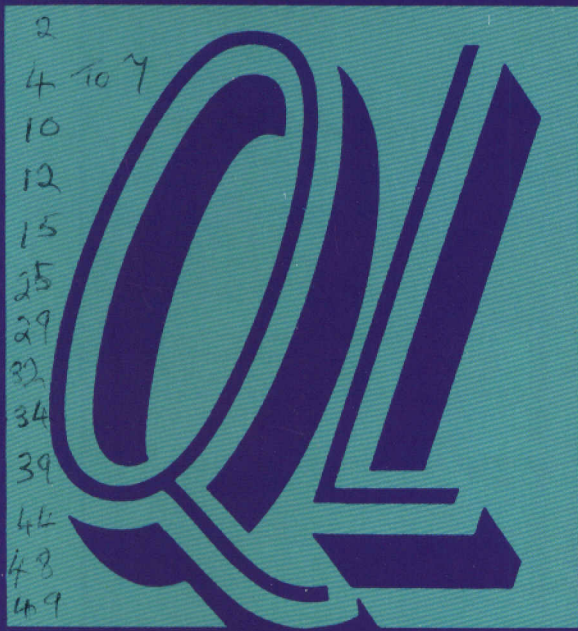


SINCLAIR

AUGUST 1991 £1.95

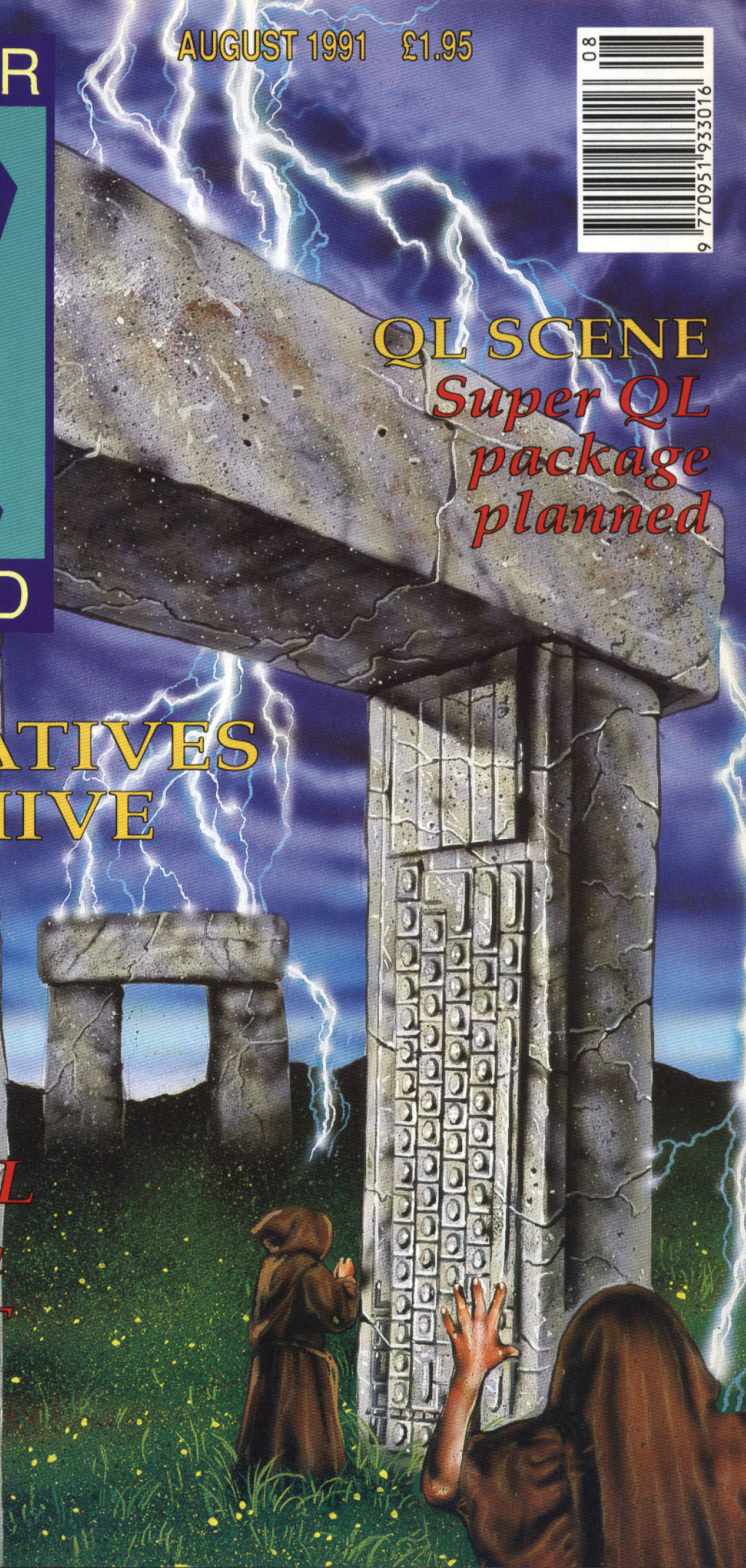


WORLD

QL SCENE
*Super QL
package
planned*

ALTERNATIVES
TO ARCHIVE

*The Merz QL
emulator on
the Atari ST*



SINCLAIR



WORLD

Editor

Helen Armstrong

Production Controller

Jayne Penfold

Designer

Jeff Gurney

Advertising Manager

Jason Newman

Magazine Services

Yvonne Taylor

Advertising Production

Michelle Evans

Group Advertising Manager

Jean Dorza

Group Advertising Sales

Manager

Lynda Elliott

Group Editor

John Taylor

Deputy Managing Director

Ray Lewis

Managing Director

Peter Welham

Sinclair QL World

Panini House

116-120 Goswell Road

London EC1V 7QD

Telephone 071-490 7161

ISSN 026806X

Unfortunately, we are no longer able to answer enquiries made by telephone. If you have any comments or difficulties, please write the The Editor, Open Channel, Trouble Shooter, or Psion Solutions. We will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2 U.K., £2.75 Europe. Overseas rates on request.

Published by Maxwell Specialist Magazines, A Division of MCPC Ltd., Sinclair QL World is distributed by IPC Market force, King's Reach Tower, Stamford Street, London SE1 9LS. Subscription information from: MSM Subscription Dept, Lazahold Ltd., PO Box 10, Roper St, Pallion Ind. Est., Sunderland SR4 4SN. Tel: 091 510 2290 UK: £21.00, Europe: £32.00, Rest of the World: £38.00.

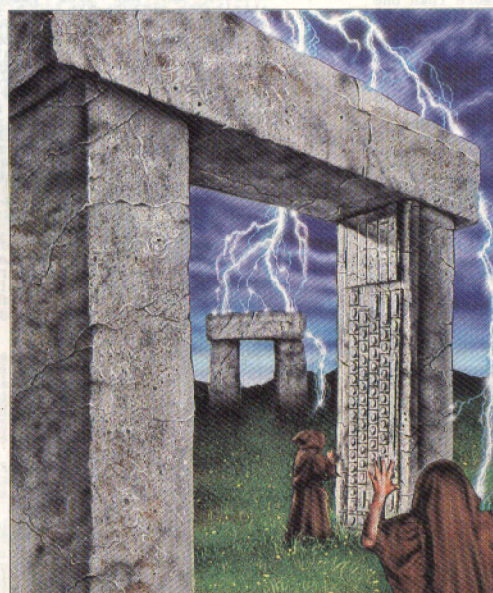
Typesetting by Ford Graphics, 8-10 Whitsbury Road, Fordingbridge, Hampshire. SP6 1BR. Tel: (0425) 655657. Printing by Cradley Print. (0384) 411411. Sinclair QL World is published on the third Thursday preceding cover date.

© COPYRIGHT
SINCLAIR QL WORLD - 1991

CONTENTS

■ ■ AUGUST 1991

- 9 **QL SCENE ● A super-system**
- 10 **OPEN CHANNEL ● Underlining the problem**
- 12 **TROUBLESHOOTER ● The QL is advancing**
- 16 **SOFTWARE FILE ● Library Manager**
- 18 **QL ON THE ST ● The Merz QL emulator**
- 19 **QL SCENE ● New from Dilwyn Jones**
- 21 **NOTICEBOARD**
- 25 **THE NEW USER GUIDE ● Part 6 – FOR and REPEAT loops**
- 29 **ALTERNATIVES TO ARCHIVE ● Flashback and Datadesign**
- 32 **DESIRES AND DEVICES ● Configuration from Superbasic**
- 34 **DIY TOOLKIT ● The MORE command**
- 39 **DBQL ● At last! Part 3**
- 40 **MICRODRIVE EXCHANGE**
- 44 **SOFTWARE FILE ● Transfer Utility**
- 46 **SOFTWARE FILE ● Discopy**
- 48 **SOFTWARE FILE ● Bargain Pack/Screen Economiser**
- 50 **INSTANT ACCESS**
- BACK COVER – SUBSCRIPTION INFORMATION**



NEXT MONTH

ONE PER DESK

A look at the current position of a close QL cousin

ARCHED

A simple database in SuperBasic

Postal Advice from Germany

Jochen Merz writes from Germany:

"Many QL users, especially from Great Britain, often ask about ordering goods from Germany. It is much easier than they expect, and I think it applies to other European countries as well. The prices in my own ads do not contain any taxed (no German MwSt., no UK VAT). The VAT will be charged by the British customs and collected by post.

"Payment is cheap as well: just send an ordinary bank-order cheque in pounds sterling, for the whole sum (goods and post/packing). There should be no extra bank charges, no expensive transfers or loss in exchange rates. Eurocheques may be drawn in DM or in the currency of the country in question, also at no extra cost.

"Money transfers between post offices of different countries is also cheap, as well as postal money orders. All these ways of payment are cheap for sender and receiver. But as soon as one German bank gets into the chain of money transfer, everything becomes very expensive!

"Apart from this good news, there is some bad news. The German Post Office (already one of the most expensive in the world) has raised its postal rates again, so that from 1st July all packages up to 2kg costs DM 12, plus registration, a considerable rise. My previous average post and packing cost of £2.10 has had to be raised to £4. Sorry."

Jochen Merz is at Im Stillen Winkel 12, 4100 Duisburg 11, West Germany.

Cowo and QL way offer a supercomputer

Cowo Electronics of Switzerland, creators of *QTop*, and Qlympic Computer Systems of Germany are looking into the possibility of compiling a 'Super-QL' – a subject much under discussion at present – to order, using the original QL board, 'the brilliant Gold Card by Miracle Systems', *Lightning Special Edition* on rom, a 1.44MB 3.5 inch disk drive and a mouse port in a modern tower housing with a 102-key MF-2 compatible keyboard. As planned the package would include the *QTop 2.0* user front and using the *QJump* extended environment. Possible options would include a second 3.5 inch floppy drive, a 5.25 inch floppy drive, and /or a 40MB

hard disk. A limited edition of 50 machines worldwide is currently planned at a projected price under £900 for the basic model.

Cowo and Qlympic are asking that anybody who would be really interested in such a package should write to them so that they can gauge how many people would buy the machine before they embark on the project, which will only go ahead if enough interested people come forward. Write to:

Cowo Electronics, 'Super QL', Munsterstrasse 4, 6210 Sursee, Switzerland or Qlympic Computer Systems, 'Super QL', Quellenweg 18, 4220 Dinslaken, West Germany.

Essex workshop update

The new date for the Essex Quanta Workshop, cancelled earlier in the year due to the untimely death of organiser Bob Gingell, is Sunday 25th August (on the Bank Holiday weekend) at the Rayne Village Hall, Rayne, Braintree, Essex from 10am to

8pm. The program is not determined yet but the venue has two rooms so they hope to be able to hold talks in one room. Further information from **Ron Dunnett at 38 Brunwin Road, Rayne, Essex CM7 5BU. Tel. 0376 47852**

Progs price cuts

The Van Auweras at Progs, Belgium, are pleased to announce a series of price-cuts in their established software. Prices from 1 June are: *The Painter* BeF 1500 (£25), *The Clipart* BeF 600 (£10), *The Painter XM* BeF 1200 (£20), *Qractal* BeF 1200 (£20), *DataDesign* BeF 3000 (£50). Eurocheques in Belgian Francs transferred to their account no. 000-1612119-76, and VISA card payment (specify card number

and expiry date) are accepted without transfer charges. Sterling cheques must include a £10 charge for transfer costs.

Progs are expecting to publish a new range of software based on vector graphics in October of this year.

Orders and enquiries to **Progs (Van der Auwera), Haacclistraat 92, B-3020 Veltem, Belgium. Tel: (Belgium) (016) 48 69 52.**

New QView

QView, makers of Minerva, are going into production on a new piece of hardware which includes a real-time clock and 240 bytes of ram, with battery backup. Called the Minerva RTC, the features which the add-on automatically recovers from ram at reset include quick start to preferred Mode, clock restored, network station number set, some Minerva enhancement optionally disabled, roms optionally soft-unplugged and one-line boot executed.

Users of rom-based extensions will be able to unplug them individually in software, so that programs that clash with extensions can be run without dismantling the hardware. Up to 128 characters of startup command may be stored in the ram – enough to enable SuperToolkit II and run a real boot file from a networked hard disk. These features can be set up via a Superbasic configuration program.

Other peripherals can be attached to Minerva RTC, which costs £65, with a discount on the full price only of £5, to Quanta members. Orders and enquiries to **QView, 29 Carnaby Close, Godmanchester, Cambs PE18 8EE. Tel: 0480 412884.**

Apologies to Digital Precision

As a result of our enquiries, we accept that articles by Simon Goodwin which appeared in the December 1990 and June 1991 issues of QL World contained incorrect and misleading matter, potentially damaging to the good name and excellent technical reputation of Digital Precision Ltd. We apologise unreservedly.

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in *Sinclair QL World*. Whether you want to ask for help with a technical problem, provide

somebody with the answer, or just sound off about something which bothers you, write to: Open Channel, Sinclair QL World, 116/120 Goswell Road, London EC1V 7QD.

Underline

Beryl Crawley's letter in QL World May 1991, brings back the memory of exactly the same problem which beset me way back in 1984/5 using an AH machine. I was then living in Botswana and had no easy access to outside help.

The key is in her first phrase 'when using *some* microdrives'. I emphasise the 'some'. I discovered that I could only eliminate the problem by formatting the microdrive(s) which had apparently become corrupted in some way. I then used my backup copies to put the data back. As an additional precaution I also formatted my *Quill* program cartridge and

re-recorded it from the master. This meant, of course, that I had to reset the Install_bas program to my requirements. After that I always kept at least two working copies of *Quill*.

Although I continued to use only microdrives until very recently, the underlining phenomenon never occurred again.

Christopher J Green
Richmond-on-Thames
Surrey

Trump

I had trouble with my 986K Trump Card, and sent the board to Miracle Systems for repair on 27th April. I received the board back on 15th May,

Editor's notebook

In Europe, Cowo and Qlympic are sizing up the possibility of assembling tower-cased QL systems using the advanced new components which have recently become available, thanks to the efforts of some of the longest-serving QL developers. It will be interesting to see what the reaction is to an advanced QL system with a projected cost of somewhat under £900 – competitive with PC/AT-type systems. It may be that many large-scale QL users feel that they are already in possession of some of the components offered and would rather assemble their own systems, but it is a bold and courageous step nonetheless.

It is little appreciated that farms were often among the first businesses to adopt home micros for accounting. No doubt this is because farmers, like scientists, often take a glum view of the reliability of wetware.

```
1 dev$='FLP1_'
2 REMark CONQUERER,
  MS-DOS/PC emulator for
  the Sinclair QL or Thor
3 REMark -----
4 :
5 REMark System BOOT program
6 :
10 TK2_EXT
20 ALTKEY CHR$(232),CHR$(233)
21 ALTKEY CHR$(236),CHR$(237)
22 ALTKEY CHR$(240),CHR$(241)
23 ALTKEY CHR$(244),CHR$(245)
24 ALTKEY CHR$(248),CHR$(166)
25 ALTKEY CHR$(234),CHR$(235)
26 ALTKEY CHR$(238),CHR$(239)
27 ALTKEY CHR$(242),CHR$(243)
28 ALTKEY CHR$(246),CHR$(247)
29 ALTKEY CHR$(250),CHR$(251)
30 ALTKEY CHR$(47),CHR$(176)
100 EXEC_W dev$&'CONQUEROR'
```

and the repair was free of charge. They have also updated the disk drivers and ToolKit 2. Very good service.

I can also recommended CGH Services and Qlympic Software. They send what you have ordered in a couple of weeks.

Pal Monstad
Norwegian All Sinclair
Association
Norway

Thor

I am the owner of a Thor XVI computer. I have a problem with the Xchange program especially with the printer setups. When I am in Archive and want to print a file greater than one page long from within Archive, the printer will print anywhere from 72 lines to 80 lines and then generates a 'Formfeed'. I even adjusted the printer driver to 66 lines but without success. I tried to enter all new data but

the computer insists on putting in 'LOPORELLO' in the printer format.

Can someone help me to get my printer to behave like it did when it was hooked up to the QL?

Another problem with the Thor is that it does not return correct tangent values. I have computer versions V6.41 and I/O processor 1.07.

Ernest Wider
Dover
USA

Capital

New capital = old capital x (1+rate/100) = credit: building societies appear to accept credits six days a week, unlike banks, which operate only five days a week in that respect.

Has anybody a program that will permit calculations of the interest earned over a number of years on a day to day basis,

as appears to be the practice? My attempts in programming such a thing have failed miserably. I need the program to adjust my income, ie the interest, over various periods for tax limiting purposes.

**RFN Patterson
Canterbury
Kent**

Mapping

It would appear that many users of PC-style keyboards for the QL are not aware that F6 to F10 on their keyboard are equivalent to Shift F1 to F5 on a standard QL keyboard, and thus have difficulty in mapping their keyboard for use with programs like PC Conqueror.

I use an IBM-style keyboard with an ABC interface and readers may be interested in my keyboard mapping. This was designed with Word Perfect in mind, and my aim was to mimic the PC keyboard as far as possible to use an easily remembered alternative. I find it easier to use than some alternative suggestions.

On my system all the function keys used on their own and all function keys used with ALT work as on a PC. This latter is possible by using the ALTKEY command in Toolkit II. Control Function and Shift Control Function do not work as on a PC. Instead Control Function is Control plus a numeral key, and Shift Function is Control Shift plus a numeral key.

For some other commands I use mnemonics:

Printscreen = Control Shift P
Insert = Control Shift I
Help screen = Control Shift H

In addition, line 30 of the boot program allows me to use the PrtSc key on my keyboard to print out a screen. In fact, the only key combination needed by Wordperfect that I am unable to obtain is Control plus Enter (hard page). Has another reader a solution to this?

As far as the function keys are concerned, my mapping should work with the Sandy, Schon and Thor keyboards. I am not sure however if all keys will work.

**Geoff Wicks
Amsterdam
Netherlands**

Care

As a result of a number of potentially fatal crashes due to overheating of the (uprated) 7805 voltage regulator, I have today replaced it with Care Electronics' QPower switched mode regulator board. It worked first time, and has so far performed without a hitch. The QL itself is running much cooler and even the power supply unit seems happier (it used to buzz to itself on a bad day).

**Alex Munden
London N16**

Tandy

Having made respectful opponents, rather than friends, with a Tandy DMP 110 and an Amstrad flat bed printer, I might be able to offer reassurance rather than solutions to those suffering interface problems. Somewhere in my files are the following:

1) Tandy DMP 110 screen dump which background-tasks on a hotkey, and allows the user to select part of the screen, copy to file, stream file to printer giving sideways dump with no lost pixels on A4. The hotkey/screen select is compiled Superbasic, the dump is assembler.

2) IBM font and filter a) QL font for character values in Ascii range, IBM font in Alt-Ascii range. (b) Compiled C filter interrogates ramdisk for a file with a funny name. When the file is found it is copied to the printer, the Alt-Ascii (IBM graphics) characters are translated to the corresponding codes. The file is then deleted. To use it, print a document, save it to the ram file with the funny name, and remember to close the file. This produces pretty documents and delightfully grotesque pictures.

I could probably dig those out - is anybody interested?

**J F Marks
Glasgow**

Editor's comment: please let me know if you are interested, and we will either contact J F Marks or, if there is enough demand, offer to print the routines.

KEYBOARD MAPPING FOR IBM STYLE KEYBOARDS

PC Key	QL Key	QL Code	Scan	Ascii	Shift	Ctrl	Alt	Ins
Pg Up	Shift Up	212	73	0	No	No	No	No
Ctrl. Pg. Up	Sh. Ctrl. Up	214	132	0	No	Yes	No	No
Pg Down	Shift Down	220	81	0	No	No	No	No
Cdtrl. Pg. Down	Sh. Ctrl. Down	222	188	0	No	Yes	No	No
Ins	Ctrl. Shift I	169	82	0	No	No	No	Yes

FUNCTION KEYS:

F1 to F5 Unchanged

F6	Shift F1	234	64	0	No	No	No	No
F7	Shift F2	238	65	0	No	No	No	No
F8	Shift F3	242	66	0	No	No	No	No
F9	Shift F4	246	67	0	No	No	No	No
F10	Shift F5	250	68	0	No	No	No	No

Control + Function keys: Use Control + number key

Ctrl. F1	Ctrl. 1	145	94	0	No	Yes	No	No
Ctrl. F2	Ctrl. 2	146	95	0	No	Yes	No	No
Ctrl. F3	Ctrl. 3	147	96	0	No	Yes	No	No
Ctrl. F4	Ctrl. 4	148	97	0	No	Yes	No	No
Ctrl. F5	Ctrl. 5	149	98	0	No	Yes	No	No
Ctrl. F6	Ctrl. 6	150	99	0	No	Yes	No	No
Ctrl. F7	Ctrl. 7	151	100	0	No	Yes	No	No
Ctrl. F8	Ctrl. 8	152	101	0	No	Yes	No	No
Ctrl. F9	Ctrl. 9	153	102	0	No	Yes	No	No
Ctrl. F10	Ctrl. 0	144	103	0	No	Yes	No	No

Shift + Function keys: Use Control Shift + number key

Shift F1	Ctrl. Sh. 1	129	84	0	Yes	No	No	No
Shift F2	Ctrl. Sh. 2	160	85	0	Yes	No	No	No
Shift F3	Ctrl. Sh. 3	131	86	0	Yes	No	No	No
Shift F4	Ctrl. Sh. 4	132	87	0	Yes	No	No	No
Shift F5	Ctrl. Sh. 5	133	88	0	Yes	No	No	No
Shift F6	Ctrl. Sh. 6	190	89	0	Yes	No	No	No
Shift F7	Ctrl. Sh. 7	134	90	0	Yes	No	No	No
Shift F8	Ctrl. Sh. 8	138	91	0	Yes	No	No	No
Shift F9	Ctrl. Sh. 9	136	92	0	Yes	No	No	No
Shift F10	Ctrl. Sh. 0	137	93	0	Yes	No	No	No

Alt + Function: redefine using Toolkit II: See boot programme

Alt F1	Ctrl. F1	233	104	0	No	No	Yes	No
Alt F2	Ctrl. F2	237	105	0	No	No	Yes	No
Alt F3	Ctrl. F3	241	106	0	No	No	Yes	No
Alt F4	Ctrl. F4	245	107	0	No	No	Yes	No
Alt F5	*Ctrl. Sh. F	166	108	0	No	No	Yes	No
Alt F6	Ctrl. Sh. F1	235	109	0	No	No	Yes	No
Alt F7	Ctrl. Sh. F2	239	110	0	No	No	Yes	No
Alt F8	Ctrl. Sh. F3	243	111	0	No	No	Yes	No
Alt F9	Ctrl. Sh. F4	247	112	0	No	No	Yes	No
Alt F10	Ctrl. Sh. F5	251	113	0	No	No	Yes	No

*Ctrl. F5 is freeze key and cannot be used in PC Conqueror

Ribbon

I bought a Panasonic KXP 1124 24-pin printer by post. The Panasonic printer ribbon supplied did not last long, and the inbuilt re-inking system on the ribbon did not appear to make much difference at first, but improved.

Then I had difficulty obtaining a further ribbon as most suppliers seem to have run out, whether of Panasonic's own make or compatibles. One firm phoned to say that they had tried Panasonic themselves, who could not supply and were going to their agents overseas. I had lots of promises. When eventually I found some ribbons I ordered more than I might otherwise have done.

I now prefer to use Text⁸⁷. This

is not an easy program to use after Quill, but is worth persevering with. Even the manuals do not reveal what it is capable of: I find that I can have What-you-see-is-what-you-get with all fonts that the printer is capable of, including double-height-double-width.

While I have had help, I find that Quill even with codes installed so that I can switch on all fonts cannot compare with Text⁸⁷, because there is no wysiwyg, and printing out is guesswork. I have at last put Quill on one side.

Of course, the printer can be driven from the 'dashboard' at the front, but using software is easier. Other facilities include being able to park a continuous roll, and put single sheets into the machine from the front.

**Stanley Hurwitz
Cirencester**

T A P R O U B L E

No amount of written claims about speed can give the full impression that a live demonstration can give, and many computer users will be — with good reason — skeptical about speed claims in general. Demonstrations can be 'rigged' to give an unduly rosy picture of what a particular item of software or hardware will do for the would-be buyer, but some of them do let you get a (rough) feel for whether or not the product in question is worth buying. One software item which is a 'must' for QL programmers is a compiler; a hardware item which may well become a must for both programmers and ordinary users is the *Miracle Gold Card*. Having recently spent a few hours looking at a screen connected to one of the early, pre-production 16 MHz Gold Cards, I now have a fair idea of what one will feel like in use. Without any doubt, it is very impressive. First, some benchmark figures.

The test program (see first box) was a simple 16-line routine, with an inner and an outer loop, and a string start-length. Not really the kind of thing the average word-processing program does all the time, but a reasonably valid test of cpu speed. The SuperBasic routine was also compiled with *Turbo*, to show what that can do. The REMark has to be removed from in front of IMPLICIT% (in the first line) before the routine is compiled. The reason for the variable start-length string is to show how SB operation gets slower with string length; there is little change in speed with string length when the routine has been compiled. Six sets of figures are given in **Figure One**: timings using the SB routine, on a JM QL with Trump Card 1 (the original, slower type), a JM QL with Trump Card 2 (current, faster type), and a JM QL with 16 MHz Gold Card, and timings using the Turboed version of the SB routine, on the same combinations of QLs and interfaces.

The runs were made with 'clean machines' — that is, nothing but the test routines loaded. The parameters used were not the same with SB as with the compiled version in all cases, simply because timings with some equal sets of parameters would have been inordinately long with SB and too short to measure with the compiled version. The difference is *that* great.

Anyone wanting to run the SB form of this routine with large values for the parameters should consider adding a few lines, to get the QL to print something to

Bryan Davis considers the huge advances made in recent upgrades to the QL.

the screen at intervals, and prove that it has not died during a test! To save you the trouble working it out, 86,782 seconds is a little over 24 hours — certainly the longest my QL has run on one job.

Looking at the figures in the second box, it is clear that Miracle's advertised suggestions for speed improvements with the Gold Card as compared to the Trump Card are reasonable. The improvement factor in these tests was about three to four times over the current Trump Card, three to five times over the older one. The merit of the current TC over the old one does not show in the shorter tests, but becomes evident as the tests get longer in duration. This is presumably because only the 'low', basic, QL memory is used when the parameter values are small. You can work percentage improvements out for yourself, if you think they are worth bothering with. Whatever the percentages, it is very clear that both *Turbo* and the Gold Card are well worth having. Note that the effect of using *Turbo* is significantly greater with the Gold Card than with the Trump Card.

Timings

Such timings don't give the *Quill* user any real idea of what to expect if he/she switches to *Perfection*, and/or the Gold Card. Using a JM QL with Trump Card, *Perfection* works fast on some operations, very fast on others. What delays there are don't occupy more than a second or so. The steadiness and responsiveness of the cursor are a pleasure to see; it moves as fast as the keys can command it, and stays virtually unwavering in shape and intensity. For those who have seen early versions of *Perfection* demonstrated, be assured that the latest version (as of early June) is appreciably faster and smoother overall and remarkably steady in behaviour for a 'young' program.

On the JS and JM with Gold Card, the speed is quite staggering. Having been

used to using *WordPerfect 5* on a reasonably brisk PC/AT, and finding the speed of the combination generally acceptable in most respects, I already knew that *Perfection* with a JM and Trump Card was a noticeably faster combination in some ways; on a JS with Gold Card, *Perfection* makes *WordPerfect* and the AT look distinctly sluggish. I'd venture to guess that the same would be true of other wp programs on the PC, and of the faster machines, such as those with a 386 processor.

Performance

Neither the Gold Card nor *Perfection* can cover up for some basic slowness in QL operation, however, so don't expect the *overall* performance of the QL/Gold Card/*Perfection* combination to be superior to *every* comparable combination of PC and wp program. Screen display speed is still a limiting factor, and disk input/output is another. It is highly unlikely that even a very smart programmer will be able to make the loading and saving of files to disk on the QL as fast as the same operations on a fast PC with decent hard disk and disk cache.

The QL hard disk is much faster than the floppy drives, but it is still relatively slow. Even with that qualification, there is still no doubt that the QL can now be placed in the same league as several more-modern computers, *provided a Gold Card is fitted*. Miracle demonstrated a go-faster board with a 68020 cpu some time ago, and they are almost certainly investigating the possibility of developing further upgrades to use either this, the 68030, or the 68040 cpu. All these should give appreciably better performance than the 68008 used in the current Gold Card. There now looks to be a real upgrade path, to encourage any wavering QL owners to stick with their favourite machine.

The external heat sink on the 16 MHz Gold Card seems to get noticeably warmer than that on the 12 MHz version, but not to the extent of giving trouble on a fairly summer-like day. The unit referred to here was in use for a session lasting nearly twelve hours and no hardware problem was evident during that time. Brief use of a later unit did indicate there may be a problem with some disk drives, as this unit failed to recognise my Mitsubishi drives although it had been working properly with NEC ones. This did not look to be a

SHOOTER

M S O L V E D

major problem to sort out, though.

One final point on this subject, this one concerning using *Lightning* with the Gold Card. In case the Miracle advert doesn't make it clear, the best version of *Lightning* to use with the Gold Card is the disk one supplied with the rom *Lightning SE*. That is, it is the reverse of the situation with the Trump Card; with the TC, you need to use the rom for maximum boost to the screen display but, with the Gold Card fitted, pull the rom out and use the SE disk version instead. With either interface, the SE version works better than the standard *Lightning*.

TaskMaster users and buyers will be pleased to know that an incompatibility between this program and the Gold Card was spotted early on and should have been sorted out well before this article appears in print. The hard disk version of *TaskMaster* is not affected by the problem. It was good to see Miracle's statement (in their advert in the June issue of *QL World*) that they have put a lot of effort into checking for incompatibilities with other products, and will endeavour to deal quickly with any that come to light. Customers will no doubt be reassured to see such regard for existing QL products stated so publicly and clearly.

Print quality

A recent article extolled the merits of the Hewlett-Packard DeskJet printer, but the printed samples did not do it full justice. Somewhere along the way, the print acquired a noticeable jaggedness. Another reader with a DeskJet has since sent me a letter to illustrate the printer's capability, and the quality is certainly comparable to that from my Epson GQ-5000 laser printer. At first glance, many people would not notice any difference. The laser output is superior, but not sufficiently so to matter for many purposes.

There is a difference of maybe £100-200 in the rock-bottom prices of inkjet and laser printers, in favour of the DeskJet. Against this, the DeskJet is apparently rather touchy about the type of paper used in it, and you do have to make sure the printouts are kept away from wetness, as the ink remains somewhat soluble in water. The running cost is not low, either; at about £13 plus VAT for an ink cartridge, which lasts for about 500 sheets of paper, you are up into the

```
100 REMark IMPLICIT% V,W,X,Y,Z
110 DIM A$(32000)
120 OPEN #3,"CON_512x256a0x0":CLS #3
130 REPEAT LOOP
140   INPUT #3,"INNER LOOP (0-32000): ";V
150   INPUT #3,"OUTER LOOP (0-32000): ";W
160   PRINT #3,"START LENGTH OF STRING (0-";32000-V;"): ";INPUT #3,X
170   START_TIME=DATE
180   FOR Y=1 TO W
190     A$=FILL$("-",X)
200     FOR Z=1 TO V
210       A$=A$&"*"
220     END FOR Z
230   END FOR Y
240 PRINT #3,"TIME TAKEN: ";DATE-START_TIME;" seconds"\\
250 END REPEAT LOOP
```

same cost range as a laser.

Dilwyn Jones Computing has sent in a review copy of its file-finder program, originally called *WinFind* but now known as *The Gopher*. The name change reflects the fact that the program can be used with floppy disks or microdrive cartridges as well as with hard disk, although its value will clearly be greatest with the latter. The program runs on a basic QL, is in Turboed SuperBasic, and comes with a well-printed and comprehensive set of instructions. Searches can be restricted to particular hard disk sub-directories, and to the first part of files (length determined by a user-specified buffer), and letter case can be significant or not.

Groups of files can be specified by entering a string of characters to be looked for in the file name; the reverse function, specifying file groups that are not to be included in a search, is also possible. The include and exclude functions can be combined for one search. The results of search operations can be printed to screen or printer. It looks to be a well thought-out program; we hope to print a review of it before too long. On a related subject, the Chairman of the QL Users' Group Quanta, Phil Borman, has provided a copy of a utility he has written to make the use of sub-directories simpler on the Miracle hard disk. Anything of this nature is welcome and I hope to have a test session with both these utilities shortly.

Readers' letters

Can anyone give **Bob Matthews** the present address of **Taylor-Made Systems**? This company used to do laser printing from Quill and *text87* files, but appears to have moved fairly recently.

The last address I had is in Kingston-on-Thames.

The closing-down of some suppliers drags on for many months, and we have received another (undated) complaint about PDQL. **Andrew Pratt** bought a Trump Card with combined 5 1/4in and 3 1/2in disk drives in early 1989, and returned the drives for repair later the same year. It was no surprise to hear he has heard nothing about the unit since then. Whether or not legal action at this time would serve any purpose is debatable, but I have heard nothing concrete about PDQL for many months. It is likely that any money available from that supplier has already been obtained through legal action by other suppliers and customers. Pratt would like to mix Korean-language characters with English ones, and wants to know if any program allows the user to design and instal founts. It's doubtful if this could be done unless he has sufficient knowledge of QL operation to dig into the operating system a bit. Quill, *text87* and Perfection all use more than one fount, and someone with sufficient understanding can modify the alternate founts used by these programs. Would anyone who has done this with Quill, say, care to offer instructions?

TK Computerware have advised that the hard disk unit supplied to **Michael Cronsten** (see comments in the June *Trouble Shooter*) was accompanied by a Customs Declaration (CP2/CP3). The reason for the £130 Cronsten had to pay to get the unit into Sweden was the 25% VAT and 3.8% duty charged on the country, on the price of the unit (about £400). VAT was not charged on the UK price, and it therefore had to be paid in the destination country. Buyers should bear in mind that VAT and Customs Duty are legal

JM QL + Trump Card 1		JM QL + Trump Card 2		JMQL + 16 MHz Gold Card	
Basic:	Compiled:	Basic:	Compiled:	Basic:	Compiled:
Iterations: inner loop 2, outer loop 100. Start length 0:					
119	0	120	-	30	-
Iterations: inner loop 2, outer loop 100. Start length 10,000:					
191	7	200	-	50	-
Iterations: inner loop 2, outer loop 100. Start length 20,000:					
256	14	251	-	72	-
Iterations: inner loop 20, outer loop 100. Start length 0:					
883	1	-	-	-	-
Iterations: inner loop 2,000, outer loop 100. Start length 0:					
86,782	32	-	-	-	-
Iterations: inner loop 20,000, outer loop 100. Start length 0:					
-	317	-	243	-	59
Iterations: inner loop 20,000, outer loop 100. Start length 10,000:					
-	324	-	247	-	60
(all times in seconds)					

tax liabilities are likely to be on expensive purchases.

Readers should be aware that complaints reported here are as stated by the readers concerned. There are at least two aspects to any story, and we do not automatically assume that everything in readers' letters is 100% correct, or that all relevant facts have been considered. What responses are received from suppliers, complaints often look to have been unjustified. It is an unfortunate function of the communication chain involved that subsequent comments upon a complaint may not be printed for several months.

TK were concerned by the original printed note about Michael Cronsten's complaint about having to pay £130 duty; they felt that the implication was that it was their fault he had to pay such a high figure. This was not what was intended, or said, in the original note, but — in case any reader misread the note — let me repeat that the transaction concerned (supply of a replacement Miracle hard disk unit) was implemented correctly and the tax charges made were correct (according to the information supplied to me).

TK can be relied upon to respond promptly when a complaint is brought to their attention. The same cannot be said about **Keyboard Products**, from whom no response has been received about several complaints, dating back to last year; all complaints concerned the 'PS/2' keyboard.

obligations — the fact that some goods come through without these taxes being charged is a function of the delivery and inspection procedures, not an indication that no taxes are payable. It seems typical that small packages pass through the system unchecked, whereas bulky ones (such as a hard disk) often get opened and checked. My own experiences with buying low-cost software from the USA is that taxes levied in the UK can be comparable with the price of the goods, because the shipping charges have been added to the price of the goods before duty is calculated.

A point to be aware of is that there can be

a reduction in the tax levied on an import, if the item concerned is accompanied by an **EUR1** or **Certificate of Origin**. TK point out that, in Cronsten's case requesting this would have saved him only 0.8% on the total of invoiced cost plus VAT, however. Note that you, the purchaser, may need to request the EUR1 or Certificate of Origin — the Post Office does not require the supplier to provide them when despatching goods (via the Datapost service) from the UK.

The above comments apply to Sweden, but no doubt similar import regulations pertain in other countries. As a purchaser, it is in your own interest to check what your

CARE ELECTRONICS

QL SUPERTOOL KIT II by Tony Tebby

Over 118 Commands:- Full Screen Editor, Key Define Print Using, Last Line Recall, Altkey, Job Control, File Handling, Default Directories, Extended Network, 16k Eprom Cartridge Version@£23.50d
Configurable Version on Microdrive@£23.50d

MIRACLE SYSTEMS PRODUCTS

OK Disc Interface (Inc Tool Kit II)@£99.64b
QL Centronics Printer Interface@£29.61d
QL Expanderam 512K ThruCard@£99.64b

QL HARDWARE

Single 3.5" Disc Drive & (Own PSU)@£105.75a
Dual 3.5" Disc Drive & (Own PSU)@£176.25a
3.5" DS/DD Discs - 10 off@£9.40c
QL Keyboard Membrane@£11.75d
Care Eprom Cartridges each@£6.11c
ULA Chip ZX8301@£15.98c

SOFTWARE 87 (State MDV or Disc)

TEXT87 (Budget Version)@£45.98d
FOUNDED 89@£15.00c
FOUNTEXT 88@£25.00c
TEXT 87/FOUNDED 89/FOUNTEXT 88@£94.94b
2488 PRINTER DRIVER@£15.00c
Upgrade to Text 87 V.3.00. Return old copy together with@£25.00d

MONITORS (Price including lead)

Philips BM7502 Green Hi-Res@£99.64a

HOW TO ORDER:

By Post. Enclose your Cheque/PO made payable to CARE Electronics or use ACCESS/VISA. Allow 7 days for delivery

QPAC II by Tony Tebby

MAKES YOUR QL TRULY MULTI-TASKING
QPAC II Main menu windows adjust automatically to size. Files, directory, view, print, delete, backup, jobs, pick, Rjob, sort, channels, things exec, wake, buttons, Hotkey, Hotjobs. Fully multitasking, allows many programs to run at once. Requires min of 256k expanded memory@£49.35b
Upgrade QRAM to QPAC II return master@£29.61b

PLEASE SEND SAE FOR A COPY OF THE QPAC II REVIEW

TONY TEBBY SOFTWARE (QJUMP)

QLFP (Micro/P disk interface upgrade)@£15.51d
QTYPE Type/Spell Checker@£30.55d
QPAC 1-Desk top accessories, calendar, alarm, calculator, typewriter, digital clock sysmon@£22.56d

ZITASOFT SOFTWARE by Steve Jones

LOCKSMITHE copies M/DRIVE - M/DRIVE@£14.10c
4MATTER + LOCKSMITHE copies M/DRIVE - DISC@£23.50c
SIDEWINDER - High resolution printer driver prints full screens or parts of screens from postage stamp size to large banners. Prints sideways, invert, scale, mirror, text insertion@£20.21c
SIDEWINDER PLUS - (for expanded QL's) includes all the features of above, PLUS multiple label printing, desktop publishing files and printer driver for 24" pin plus printers LC10 and Jx80 colour printers. (Please state 3.5" disc or M/D)@£23.50c
Upgrade to Sidewinder Plus@£8.46c

QPOWER REGULATOR

Switch mode power supply, QL runs cool, stops, lock ups (due to voltage drop). Helps filter noise out. Simple to fit (no soldering)@£25.85c

MULTI COLOURED PRINTER RIBBONS

Gold, Silver, Magenta, Orange, Purple, Brown, Green, Blue and Red.
Citizen 120D/Swift@£7.99c
Cannon 1080A/1156A@£9.87c
Epson FX80/LQ400/LQ800@£5.17c
Epson FX100/LQ1000@£7.99c
Panasonic KXP 1080@£8.46c
Star LC10@£7.52c

READY MADE LEADS

RGB QL TO PHONO@£6.11c
RGB 8-6 PIN@£7.52c
RGB 8-7 PIN HITACHI@£7.52c
RGB 8-7 PIN FERGUSON@£7.52c
RGB 8 PIN TO SCART@£11.75c
6WAY PCC TO 25WAY 'D'@£9.87c

24 Hour Order Line 0923 894064

OPEN
9am-5pm Mon-Thu
9am-4pm Fri

CARE ELECTRONICS

DEPT QL, 15 HOLLAND GDNS
GARSTON, WATFORD,
HERTS, WD2 6JN.
Tel: 0923-894064
Fax: 0923-672102

Please add carriage

a=£11.75 b=£3.76
c=No charge d=£2.35

New disk programs from Dilwyn Jones

Three new programs from Dilwyn Jones Computing should be on the market by the time this is published, as well as an updated version of the popular *Home Budget*.

All three programs are concerned with tidying and organising disk and cartridge collections.

Super Disk Labeller is a program for printing disk labels to size, the label is constructed on the computer and then printed to the correct format.

Filenames can be listed in tidy columns across the label, in a choice of six layouts. The lists are printed in small text to fit a reasonable number of filenames onto a standard disk label. There is also the option to list file sizes, types and update dates if required. Label headings can be added in a different size if desired.

Filename lists can be sorted into alphabetical order, or sorted by suffix or sub-directory. The user has a choice of which filenames are listed.

The label is constructed by inserting the disk to be labelled in the disk drive, and operating the relevant keys to initiate the label-making routine. There is a short-cut method if you do not want to use all the options.

The program works with 9- and 24-pin Epson compatible printers, and the printer-driver is user-editable. Text size and line spacing are under user control.

The program works with 9- and 24-pin Epson compatible printers, and the printer-driver is user-editable. Text size and line spacing are under user control.

The program as supplied is configured for use with 3.5 inch disk labels as supplied by DJC at £2.50 per roll of 100, but can be altered to use 5.25 inch labels or address labels as needed.

The menu-driven program

SDL_QLW 780/1440 sectors SUPER DISK LABELLER	
9PIN_PROP_DRIVER.dat	(a printer driver)
9pin_DRIVER.dat	(a printer driver)
24PIN_DRIVER.dat	(a printer driver)
24PIN_PROP_DRIVER.dat	(a printer driver)
BACKUP_bas	(program to make a backup copy)
BOOT	
SDL_code	(extensions used by the program)
SDL_PRINTER.dat	(default printer driver)
SDL_task	(the Super Disk Labeller program)
UPDATES.doc	(last minute notes about the program)
KXP1001_driver.dat	(a Panasonic printer driver)

An example label in small print, with explanatory text added using Super Disk Labeller.	

needs at least 128 Kbytes of extra memory to run and comes with a 20 page manual including sample printouts and instructions. Disk insert labels can also be printed. Super Disk Labeller costs £10.00 on 3.5 inch or 5.25 inch disk. Postage and packing is extra to customers outside the UK (see below).

Super Disk Indexer by Imre Dominik is an aid to cataloguing programs and files. The program asks for the number of the disk (or cartridge) in the drive (allocated by the user up to 999) while reading its directory. It then puts the filenames into a list. You can sort this list into alphabetical order if desired, and incorporate it into the main index. Unwanted filenames can be removed manually. The index can be sorted by alphabet or by disk number as preferred.

Filenames can be searched by whole or part names - exact or vague searching - and will list all occurrences of a filename in the collection. The menu-driven program comes with a full manual and requires at least 384K of memory. It can be supplied on 3.5 or 5.25 inch disk or microcassette, although disk is recommended. Super Disk Index cost £12 inclusive of UK postage.

The Gopher by Norman Dunbar

is a file-finding utility: given details of the text required, it will hunt through all the files on a disk (with a by-directory option for hard disks), reporting the names of all files containing the required string. Requesting a search by file extensions is one way of narrowing the search area if preferred - DJC point out that there is a tremendous flexibility in searching, with up to eight parameters for searching available, with defaults so that the more complex parameters are only accessed if needed. Whole or part files can be searched.

The Gopher will dig for files on floppy disk, hard disk, microcassette or even ramdisk. Available on 3.5 or 2.25 inch disk or microcassette, with a printed manual, the program costs £12 including UK postage.

The new update to the *Home Budget* by Joe Hattke incorporates changes resulting from the Chancellor's Budget Statement this spring. The tax calculator in particular has had the rules brought into line with the new tax system.

Copies supplied since April 1991 have automatically had these changes incorporated, but earlier users can, for a limited period, upgrade for a fee of £3 post inclusive by sending their

master disk back to DJC. A few minor bug fixes have also been incorporated. Please mark your envelope 'Home Budget Upgrade' when returning your disk.

In addition to the *Home Budget* upgrade and the *File Tidy* series, DJC have recently released three general interest programs, *Question Master* by C B Storey (£10) is a multiple-choice question and answer program, easy to use and with a sample file included. You can make your own question and answer sets, or try *General Knowledge Quiz Questions* (£5) and *Classical Music Quiz Questions* (£5), also from DJC, *Screen Economist* by G Estournet (£10) will turn off your monitor display if no key is pressed for a period of time, to help protect the screen from static phosphor 'burn'. Single-key commands select a time delay from one minute to four hours (with a default of 10 minutes) and another 'hit any key' command will switch the display back on. A function will display the current delay time as a reminder if needed.

Trans 24 by Ralt Rekondt (£10) will convert graphics files generated for 9-pin printer to a 24-pin printer driver format, if the graphics program concerned is capable of printing to a file. In this way packages which only have 9-pin drivers can use the superior graphics capabilities of 24-pin printers without resorting to super-light printing. The program attempts to compensate for the 24-pin line spacing so that the output will appear as close as possible to the normal 9-pin size.

These three programs will run on an unexpanded QL, and are available on 3.5 or 5.25 inch disks or microcassette.

All orders and enquiries to Dilwyn Jones Computing, 41 Bro Emrys, Tal-Y-Bont, Bangor, Gwynedd LL57 3YL, UK. Tel: Bangor (0248) 354023. Prices are inclusive of UK postage. Overseas customers please add £1 for postage to Europe, £2.50 elsewhere.

SOFTWARE FILE

INFORMATION

Program: QL Library Manager (V2.0). 256K Memory needed.
Supplier: Ergon Development
 c/o Davide Santachiara, Via Emilio de Marchi n.2, 42100 Reggio Emilia, Italy. Tel. (+39) 0522 70409.
Price: £25 (£27 on mdv) + £5 p+p.

QL LIBRARY MANAGER

QL Library Manager (QLM) is the second program written by Ergon.

It is written in Turbocharged Basic and is the first program to use Ergon's own Basic System Development Menu system.

The program itself looks like a simple idea: it enables a QL user to keep a 'library' of all his or her favourite Basic routines in the form of procedures or functions. This library is kept on disk, so that the user can access the routines at a later date for inclusion in a new Basic program. The only problem with keeping a library of such routines is that when you later want to extract a procedure out of an earlier program, you have to wait a long time for the library to load, find where the procedure is stored, SAVE this part of the program to disk and then merge it with your current program.

Extract

A library manager's job is to extract all desired routines easily and quickly. Turbo users will already have a simple library manager which carries out a similar job. The program supplied with Turbo is okay for working with the supplied Turbo library, but can cause problems with other libraries because the user needs to keep a note of all the routines which are called by those he wishes to extract and then ensure these sub-routines are also extracted. This program is much more

Rich Mellor meets a package which stores, catalogues and accesses SuperBasic routines.

intelligent, works a lot more quickly and should be welcomed by anyone who develops Basic programs.

The program uses the normal Ergon copy protection of the user's name and address (plus a secret number), and on loading, this is displayed along with a title screen giving information on the program. Pressing a key, shows you the main screen of QLM. This is in the form of a large message window, a small information window in the top right corner and a menu panel.

All of the menus used by the program will seem familiar to a Qpac II user (although the QPTR system has not been used – this cannot be compiled with Turbo). Each option can be se-

lected by using the cursor keys plus the space bar, or by pressing a letter shown in front of each option. There are however certain differences from normal Qpac menus:

a) there is no scroll bar (or indeed any indication that not all of the options are actually displayed on screen) – as you attempt to move the cursor keys past the top or the bottom of the menu, the options will scroll up or down; and

b) the position of the menus is fixed (although this does not affect the program in any way).

The menus also use certain function keys (CTRL and P/A/S). CTRL A selects all of the options; CTRL S toggles all selected and unselected options; whilst CTRL P affects whether

or not the previous two keys apply to all functions available on the menu, or just those which are presently visible on screen.

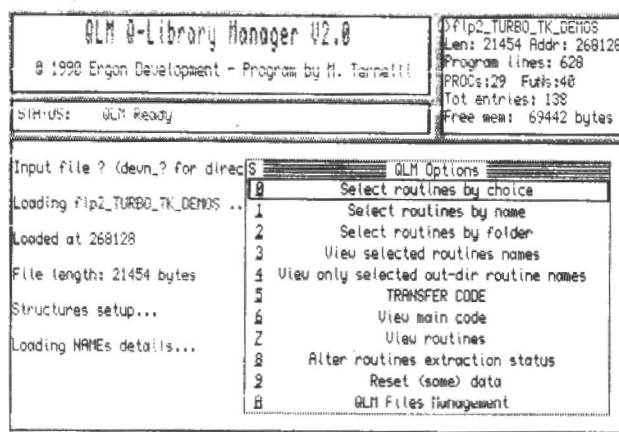
The menus are very easy to get used to, and provide a very user-friendly method of using the program. As a further aid to the user, the program attempts to select all of the most obvious options on entering that menu. This helps to cut down the number of key presses needed and also means that much of the program can be used almost without any intervention by the user.

Menu

On loading the program, the first menu offers the options Load a File, Enter Multi-File extraction, Define Multi-File menu, Add a file to Multi-File menu, Remove files from Multi-File menu, Rest Multi-File menu, Merge files, ADDNUM/RENUM a file.

I shall look at the multi-file options later as these are amongst the most complex and the most difficult options to get to grips with.

The other options on the menu allow the user to load one 'library' file from which to extract the necessary routines, merge two libraries together, add line numbers to a program or renumber a program (the latter however does not renumber any GOSUBs or



A file loaded and ready to sort out desired routines.

GOTOS). Choosing a filename could not be easier, since if you are unsure of the filename, entering 'flp1_?' will produce a menu containing the directory of flp1_., rom which you can then select the name of a file using only the cursor keys and space bar. While using this menu, should you be uncertain as to which file you want, pressing F4 will copy the file currently under the cursor to screen. Once the filename has been chosen, the user must state whether or not this file has a _QMW file with it.

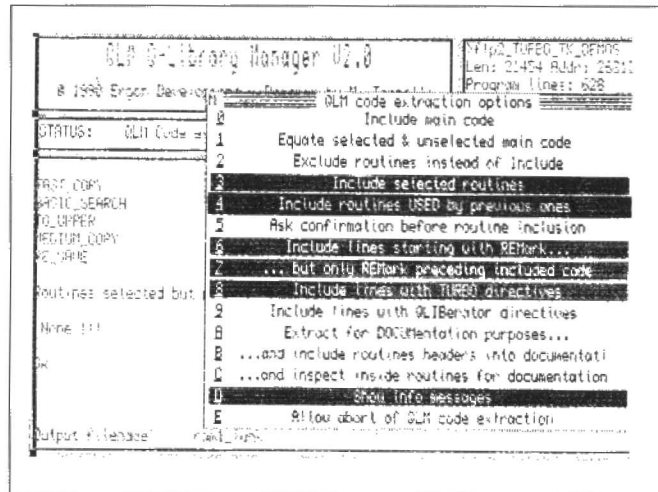
Accessing

The '_QMW' file contains data for accessing all of the different routines contained in a library file. If one is not present on the disk, QLM will set one up, which can then be later used by QLM. This data file ensures that routines can quickly be found and accessed by QLM, thus making the management of a library much easier. This is quite an intelligent routine, in that it can distinguish between names of routines in strings or REMarks, and will also issue a warning if a routine is defined twice. As with most of the other slower file access routines, the user can decide whether or not to have the output shown on the screen. If you answer no, then nothing is printed to the screen, ensuring that if you are using the Qpac multitasking system, you can do something else while the file is processed.

Once the file has been prepared in this way, the user is presented with a menu which allows the parts of the library file which are to be extracted, to be defined in several ways:

i) The first of these is by using a sub-menu which contains all of the names of the available routines. You can then select the chosen routines using the cursor keys and space bar (and view them by F4 if you are unsure).

ii) You can also choose to select the routines by entering their names. If you are at all uncertain, you can choose to view the procedures/functions in the library, or any main code which surrounds them (generally REMarks, but also possibly main parts of a program which would be executed if the program was RUN).



Desired routines ready for extraction from the library.

iii) Should there be any routines which you always tend to extract together (for instance a save routine and a load routine), then a 'folder' can be created to hold the names of these routines. This folder can then be included on subsequent extraction processes, so that the user does not have to keep selecting all the names. This folder can be saved to disk along with the rest of the library, and can even refer to other folders. This provides a tree structure for the library which enables cross-references to be implemented and thus to improve the usefulness of a library. It should also be noted that should a folder refer to any names which are not present in the current library file, these names are highlighted so that they may be later extracted from the correct file.

Powerful

I previously mentioned QLM's ability to access MULTIFILES. This is a very powerful feature of the program, and allows the user to merge routines which appear in several library files. The option is quite easy to use, since you only operate on extracting routines from one file at a time. However, any routines which are not present in the given file, will then be searched for in the next library file and so on. This ensures that library files can be kept relatively short and therefore not be impeded by their speed of loading should you wish to update one of the routines.

Once you have reached this

far, you are ready to extract the given routines. This (as with all of the other options) is not as simple as one may think. There are lots of different options which can be used to ensure that everything a user needs is extracted in one fell swoop. On entry to this menu, the user may well find that all of the necessary options have already been chosen for him or her by the program. However, should the user so desire, he or she can also choose to include 'main code' (ie parts of the program not located between DEFINE PROCEDURE/FUNCTION and END DEFINE), include all REMarks in the library file (or just those which appear the line before a selected routine); or include Turbo and/or QLiberator directives.

Having chosen how the extraction is to be performed, the program will pick out the selected routines, and if not told otherwise, will check to see whether the selected routines make calls to other non-selected sub-routines, and will include these as well. It need only take a matter of minutes from deciding which routines to include, to having them loaded ready to be MERGED with your new program.

Manual

But the package does not end here. Should you wish to write a manual for your new program, it is easy to forget to include information about some parts of the program. To this end, the Extraction menu also has an option to create documentation. It does this by extracting all REMark lines,

compiler directives, names and parameters of routines used by the program. The user can choose how much of this information is extracted. The output can then be used to create a user manual, or a more technical programming manual on how to use various Basic routines.

The manual supplied with the program is very good, although it needs improving in one or two parts where the translation process appears to have broken down (I understand this is being looked at). Thankfully a printed manual is supplied, although once the user has used the program once or twice, there should be little need to refer to the manual.

Overall, the package is useful for SuperBasic authors, and should enable new Basic programs to be created quickly, using experience and routines already created for earlier programs. It will certainly entice the user to keep a library of all his or her own routines, rather than re-inventing the wheel every time he embarks on a new project. Programs like this are essential to anyone who wants to maintain a certain 'feel' to their programs, since it means that all routines used within a suite of programs will be the same.

Windows

As a little aside, there is another small program supplied with the package. This is a window definer program which enables the user to open as many windows as he wishes around the screen in different paper and ink colours, by using the cursor keys and the space key. Once you have selected all of the windows your program will use, a small Basic program is created which will open the windows as you have set them out.

When sending for the package, cheques must be made out to D. Santiachiar. Please note that you must add a £4 conversion fee if the cheque is made out in sterling. This may seem expensive, but if you send a Eurocheque in Lire, the cheque fee is only £1. There are discounts if you buy more than one program, and the cheque conversion costs only apply once for each cheque.

The QL on the ST

With the advent of Miracle's Gold Card a new quantum leap has been made in the weird and wonderful world of the QL. The gestation period has been a long one, but finally the spectre of a modern version of everyone's favourite computer seems to have been transformed to reality.

There was a time when the QL's hopes and future looked so bright and full of potential. I for one knew when the first ads came out that this was the machine for me. I migrated from a very suspicious attitude to computers – the Big Brother syndrome via the Hewlett Packard HPCX programmable calculator, to the QL back in 1984 or so. Despite the blurred images on a portable tv, the ever-whirring microdrives, the obscure crashes, arguments with the printer, the Medic Saga, and . . . you name it, I always had the faith that the Quaint Lump was designed to work, and that with perseverance it – eventually – would.

All along the way rumours abounded about upgrades – faster, better, cheaper, The Thor, Futura, transputer cards, somehow none of this seemed just right in the same way as the QL when it first appeared.

When the Atari QL Emulator arrived I again had that inkling that this was somehow spot on. I rushed in, about two years ago now, but, looking back, I discovered that there was virtually nobody else there. I attempted a few articles/reviews on the machine, but they all seemed too, too enthusiastic to send in. In this review – rather more practical than technical – I hope to show a more balanced view.

To begin with, admittedly, there were hiccups. It was the QL tale all over again. However, things have settled down and what we have here is the QL as it could have been had its development with Sinclair not been untimely cut short.

First of all you have a stylish grey box with a built in 720K disk drive, an excellent, separate keyboard (this is the Mega ST), and a mouse. At the rear are an array of ports: dma (hard disk, co-computers), additional disk drive, monitor, MIDI in, MIDI out (musical instrument digital interface), parallel, power, and serial. There is also an on/off switch, and the reset button at the rear. The 128K rom port sits on the left side of the cpu. Inside is an 8MHz MC68000 16/32bit processor, up to 4 Mb of ram, battery backed clock, floppy

Per Witte is a happy and successful user of Jochen Merz's QL emulator on the Atari ST.

controller and assorted chips, not all of which are currently relevant to the QL. The emulator card has to be mounted inside the machine. It involves removing the Shifter chip, plugging in the emulator pcb, re-fitting the Shifter, bending a few pins on the GLUE and, finally, soldering seven wires on to various points.

Once the machine is fitted out and connected up it can behave either like an Atari ST or a QL. The emulator software starts up as a normal Atari program, and then takes over the machine completely. At the familiar F1/F2 screen it is wise to boot up the QL software that is supplied with the emulator. This includes Tony Tebby's *Toolkit2*, new drivers for floppies, hard disks, ram disks, ser, par, prt, pipe, null, keyboard, new screen drivers (Pointer Environment), Things, mouse drivers, and a host of utilities: conversion of disk formats (Atari, IBM), menus, TRANslates, Hotkey system II, and much more.

In use the QL-emulating Atari works like a very fast (two to three times) QL, with no hitches and, in my experience, very few compatibility problems. True, you cannot access microdrives directly, so those naughty programs that are microdrive

bound, such as Psion Chess, and the likes, are, unfortunately not generally available on this machine, though a number of these programs can be (and have been) hacked to run without the microdrives. Also QL-specific hardware add-ons are of course not possible. This is not a problem for me, but for some it may be a major drawback.

For the sensitive-of-hearing there is another hitch: the fan (on the Mega STs only). It is not extremely noisy, but it is there, humming away at all times. There are ways around it. The way I have gone is to buy a hard disk – because it makes so much more noise, you can hardly hear the fan – a rather drastic cure.

Advantages

The advantage of this particular system is that you have a fast, comfortable machine, with lots of memory, reasonably priced peripherals, reasonably good support, 99% QL software compatibility (JM/JS and equivalent), excellent bundled software; a 'free' Atari computer with access to accelerators, other emulators (Mac, IBM, Beeb, CP/M, Atari 8bit, and possibly others), and some very good software (music, dtp, graphics, languages, comms, PD, games), a very versatile machine indeed. (I know the Amiga is a more desirable machine from a QL user's point of view, but it is just because the Atari is so 'plain' that it makes a good emulator host.)

There are some disadvantages. Hardware incompatibility may be a serious



disadvantage for some users, though there are ways to work around some of the problems: roms can be catered for with a special adaptor, I believe, otherwise they can be loaded into ram with a special command ensuring that they occupy the correct position in the memory map. Most software can be transferred from microdrive to disk (microdrive emulation is supported for those who cannot or will not convert their programs). Hardware that is addressed via the serial or parallel ports (eg modems, printers) should work, while some types of hardware, such as memory expansion, disk-drive and parallel interfaces are made redundant. There is no network on the emulator. This may come in time, using the MIDI or serial ports.

I have kept my old QLs – they are too much of a bargain. If I want to play the odd chess game I do it on the QL, as before.

I would like to spend some time on the software that comes with the emulator, as it could be considered integral to the system.

The first major difference that one discovers, after the initial glee over the slick speed and quality of the machine, is the Pointer Environment (PE). This system, for which Tony Tebby is responsible, turns multitasking into what it should really be: non-destructible windows with the saving of contents during task-switching, the ability to use the mouse to select from menus and perform other tasks, and the provision of a configurable user interface that uses standard tools to perform standard operations.

The idea is that each task is attached to a main window. Within that window the task's interaction with the user takes place. Other windows owned by that task can be opened within the main window, to present a menu, for example, or to display a message. The main windows are 'stacked' one on top of the other, giving the illusion of a desk piled with 'live' sheets of paper. One can cycle through the pile using ^C, which is a bit like shuffling through the pile on your desk to find a certain sheet; or you can use Hotkeys, which simply call up the sheets you want straight away; or you can use the mouse. If you can see even just a tiny corner of the sheet you require, move the pointer to it and click. Presto! The required sheet is brought to the top of the pile, awaiting your pleasure.

Moved around

Those programs that are specifically designed to work with the PE (and there are a growing number of them), can usually be moved around the screen. For instance, you are typing in *Quill*, call up a calculator, move it somewhere else to prevent it from obscuring the text/data you are working on, do your calculation, stuff the result in the stuffer buffer, click on *Quill*, and squirt the result back into your document. Windows can also be resized

if appropriate, simply by using the mouse. Old programs, such as the Psion Four, cover the whole screen, losing some of the versatility of the PE. Still, they can be used to advantage within the system.

Most of this is transparent to the user, who can continue to use the machine as normal, though with less hassle. A disadvantage of the PE is the considerable amount of money that is taken up with screen saves. Part of this is offset by the technique of code sharing, by which a single copy of a program can run as multiple tasks, only taking up additional space for data. Other task swappers can also do this, without the memory consumption of screen saves but to be quite frank, if you have got the memory, there is no real contest for efficiency and ease of use. Hotkeys are not only used to load, wake and pick up tasks. You can also tie SuperBasic commands to hotkeys; or a series of key presses relevant to other programs, such as Abacus. This gives a virtual macro-capability to the QL. A very nice touch indeed.

In this way it is possible to tailor-make your working environment to what suits you – and the task in hand – best. It is not easy to come to grips with some aspects of the system, especially not for those who prefer just sticking a disk in flp1_ and letting the program take control (but they should really have bought a PC in the first place, shouldn't they?). The pointer environment and hotkey system is available for the Sinclair QL too, but it really comes into its own on a fast machine with a megabyte of memory or more. These are not major requirements when you consider what other operating systems (Window, PS/2, Unix) require just for breathing space.

The manual that comes with the emulator is adequate to get you started. Most of the really powerful features will remain hidden for all but the most determined hacker, as they do not seem to be documented anywhere in the manual. "What you don't know won't hurt you" may be true in some respects, but trying to write programs yourself that must be compatible with features that are not documented could cause frustration.

I live and work in a community with people with learning difficulties. In addition to the toil of everyday life I work with administration, accounting, records, applications and so on. In the beginning all our administrative chores were done by hand, the methods evolving over the years as the Community grew and established itself. By the time I joined, about five years ago, the paperwork was increasing, and four people were tied up in the office, part time. I brought my QL with me and used it as my personal computer. Little by little I took on more of the administration, especially the book-keeping side of things.

For a while I flirted with the idea of purchasing an accounts package with a

computer to go with it, if necessary, but I didn't find anything that I felt would let us do things our way: we were expected to conform. I find office work a bit tedious. Luckily I found computing extremely interesting, so by combining the two I found that aspect of life quite tolerable, and got the job done at the same time. The result is that many of the tasks that need to be done can be done the way that suits the style of the Community, and the preferences of those who are doing them.

Accountancy

I use *Quill*, *Abacus*, *Archive*, *QPAC2*, and my indigenous accountancy program *ACC*, every day. Additional help programs such as *QPAC1*, *QTYP* (the spell checker) and assorted utilities to make life easier are also in constant use. One of my QLs got a new keyboard and has been used by our secretary for the last year or so. A few months ago we decided to go the whole hog and the Community purchased its first computer: an ST Mega 2 with emulator. Secretary Cathy is mightily impressed, and she's getting the hang of things now. Yes, it has taken some time, partly because she had to unlearn everything she knew about the other computer, not really coming to grips with multitasking, also because previously there was very little room to manoeuvre. Between us we now have an Epson LX80 dotmatrix printer and a Hewlett Packard Deskjet+. My Mega 4 has a hard disk, which is convenient, albeit noisy.

When two or more users are working on similar data, it would make sense to have some kind of networking capability. At present we have no relevant programs that can take advantage of such a capability, perhaps because the QL's net, though very flexible, is too slow for professional work. I can never get enough speed! Sorting, searching and calculating should really zip along.

My main wish, though, would be for quieter and smaller machines. Apart from the latter, I think I've got everything I need at the moment. As for the rest, I'm perfectly happy to sit back and wait for evolution to take its course, provided that involves keeping the QL alive!

The emulator can be obtained from a number of sources. My first one I bought from Futura Datasenter in Norway. The Mega 4 I got from Strong Computer Systems who installed the card and kept me in warranty. They were very helpful and bent over backwards to keep me up and running. Unfortunately they don't seem to be in the QL market any longer, and so I took my business to Jochen Merz in Germany where, despite the distance, I have had very good service and support. Tony Tebby, who has written most of the propriety emulator software, and Jochen Merz, ensure that the emulator is of excellent quality and are continuously up-



dating the system software.

Ataris with the emulator ready installed can be obtained from Jochen Merz, and also some British dealers (look out for their ads). The Community machine I got from Evesham Micros, who are not too far away from where I live. They installed the Merz emulator for me and promised to honour my warranty provided the emulator was not to blame.

You need to buy an Atari for £250 (512K)

to £800 (4Mbytes), and the emulator card, including software for about £200. (Important: check with your emulator supplier which models are NOT suitable before you buy!) Installation can either be done by the brave hobbyist, for about £25 at Evesham Micros, and possibly for about that elsewhere. You may be able to use your existing monitor. For the rest, disk drives, most software, printers and paraphernalia should still be compatible.

How does this compare with other options? Take an ST Mega 2 at £615 (June 91). The total for machine, emulator, software, floppy adaptor, and installation, assuming re-use of everything else you need, is approximately £850 including VAT and p&p, to a round figure.

The Gold Card from Miracle Systems is £375. This may be all you need. You may however want a mouse (£50 incl. interface, excluding RTC), a keyboard (£100 interface + cheapish XT keyboard), installation of the above (£25, say), and an additional disk drive (around £50). The total comes to about £600.

Thors, if obtainable at all, are more expensive than a corresponding ST.

Stay with your existing system and it costs you nothing but your time.

What it costs depends on the difference between your needs and what you already have. Obviously the Gold Card represents good value, and of considerable significance, I think, is that more money then goes in the direction of people who support the QL (rather than Atari Corp). The extra speed of the Gold Card is considerable, and it will cost you £200 extra to achieve that on the Atari. The Atari option gives you three(-ish) computers rather than one, don't forget (2Mb Atari/QL plus your old QL), so it's not a bad deal either. There are other pros and cons either way. Whatever you do, keep QLing for general computing it's still among the best.

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12
4100 Duisburg 11 - W-Germany
Tel./Fax 0203/501274

NEWS: The current emulator software is level D (named pipes complete Extended4-Screendriver & more). There are new updates on QD (e.g. PRINT always works now), on Menu etc. **NEW PRODUCTS:** The QDOS Reference Manual, EASYTPR II and QPrommer. Finally, credit cards welcome now!

QL-Emulator (Hardware, QJUMP's Drivers AtariDOS & utilities) **£166**
EPROM Cartridge for ST (makes machine autoBOOT), switchable **£33**
Centronics Cable for ST: 2m £6.50, 5m £10.50 **MIDI Cable** 2m **£3.50**

ST-Monitor-Cable: Scart £9, Cinch £11, Amiga 10845 £13
Floppy Adaptor to connect QL-Disk-drives to the ST **£9**

QDesign — A new graphics program which uses the Pointer Environment. Very easy to use, with extras never seen before in a QL graphics program: Editable area ranging 512x256 pixels up to 2880x2880 (depending on your RAM) You can even load monochrome 640x400 ST pictures and use them. Page View, Scaleable Vectorfonts etc. **£38**

QD III — Very comfortable text-editor, running under the Pointer Environment. Many features (e.g. Menu, Scrap) **£38** Upgrade from VI or VII **£10**

QPTR — Re-released & Updated! Toolkit for Machine Code and BASIC Pointer Environment Programmers. Revised manual & keys/macros updated. CONFIG is also explained. **£30** Update (new manual & disc) **£14**

QMENU — Many QD Users will know it: the Menu Extension. Now with more facilities, even real subdirectories are treated in a new file-select window. Programming instructions and examples for SuperBASIC and machine code. A Scrap Thing is also implemented now! **£11**

QPROMMER — Menu-driven, very comfortable EPROM-programmer software which allows you to use the ATARI Junior-Prommer under

QDOS. **£19**

EASYTPR II — Create your own Pointer Environment menus & sprites and use them in your own SuperBASIC or m-code programs. Many users waited for this package, as creating Menus was not easy, until now! Put all the items you need on screen, that's it! Supports all the sub-windows and uses the Menu Extension! Now even complete application-sub-windows may be created! Many examples! **£49**

EASYTPR II — Budget version **£29**
Upgrade from version 1 **£10**

NEW! NEW! NEW!

QDOS Reference Manual — This book is a must for all m-code programmers. It explains how to use QDOS, all traps and vectors, the Thing System, the HOTKEY System II and much more. It points out which features work on a QL, an Emulator and how to write compatible for future operating systems. DIN A5, 170 pages. **£30**

DIAMONDS — A "just one more go" game to keep you up at night. **£11**

QLQ £21 **Thing & EPROM Manager** **£18.50** **Firebirds** **£10**
Brain Smasher! **£12.00** **QSUP£26** **Arcanoid II** **£10**
Ion Gold and Doppel Ion **£10** **Super Games Pack** **£25**

Pack of 10 3.5" Discs MF2DD incl. Labels **£3.10**

Keyboard Interface (for IBM-type PC/XT keyboards) **£41**

QIMI — QL Mouse Interface with Real-Time-Clock **£41**

High Quality Mouse for QIMI or ATARI ST **£27**

QIMI without RTC (for Gold Card) **£37.50**

Adaptor AMIGA mouse to ATARI or QIMI or v.v. **£2.50**

Please add **£4.00** for postage and package.

All programs except Arcanoid, Firebirds & Ions need memory expansion. All prices excl. V.A.T. E&OE



THE NEW USER GUIDE

Part 6 of the New User Guide builds up FOR and REPEAT loops which allow processes to be repeated automatically — and embedded within other repeated processes — under the programmer's control.

SECTION SIX

A long time ago I was shown how to prepare a car for display on a garage forecourt. 'Here is a typical car. Wash here, polish here, steam clean the engine compartment, brush out the boot.'
'Now,' instructed the forecourt owner, with a grand sweep of his arm, 'do the same for all of those.'

So far in this tutorial there has not been a concise method of giving the computer similar instructions to repeat a sequence of commands. The nearest we have come to it is the multiplication table program where a GOTO statement has redirected the computer back to a previous line:

```
100 LET N = 2
110 PRINT N * 4
120 LET N = N + 1
130 IF N = 13 THEN STOP
140 GOTO 110
```

This routine undoubtedly works, but it is more than a little cumbersome. It takes up five lines, it does not show at a glance what is going on and the line which should be most prominent — the instruction to multiply N by 4 at Line 110 — is lost among the other statements.

The essence of the routine can be summed up using one word for each statement: initialise; calculate; increment; test; loop. SuperBasic very helpfully has a single statement which combines all but the calculate stage, shortening and simplifying programs.

The key to reducing the above five program lines to just two is to recognise that we are here dealing with a range of values for the variable N. The routine could be written in English as 'For each value of N from two to twelve, print N multiplied by four'. SuperBasic would be criticised mercilessly if it were so verbose, so the programming equivalent of the first part of the sentence is:

```
100 FOR N = 2 TO 12
```

In Basic this forms a complete statement, and so it can stand on a line by itself. The second part of the sentence, the action which must be repeatedly carried out, follows on the next line:

```
100 FOR N = 2 TO 12
110 PRINT N * 4
```

In English we can use a full stop to indicate that the sentence is finished. SuperBasic has to allow for the fact that the actions being carried out might occupy a number of lines, and so a special statement is needed to complete the FOR loop:

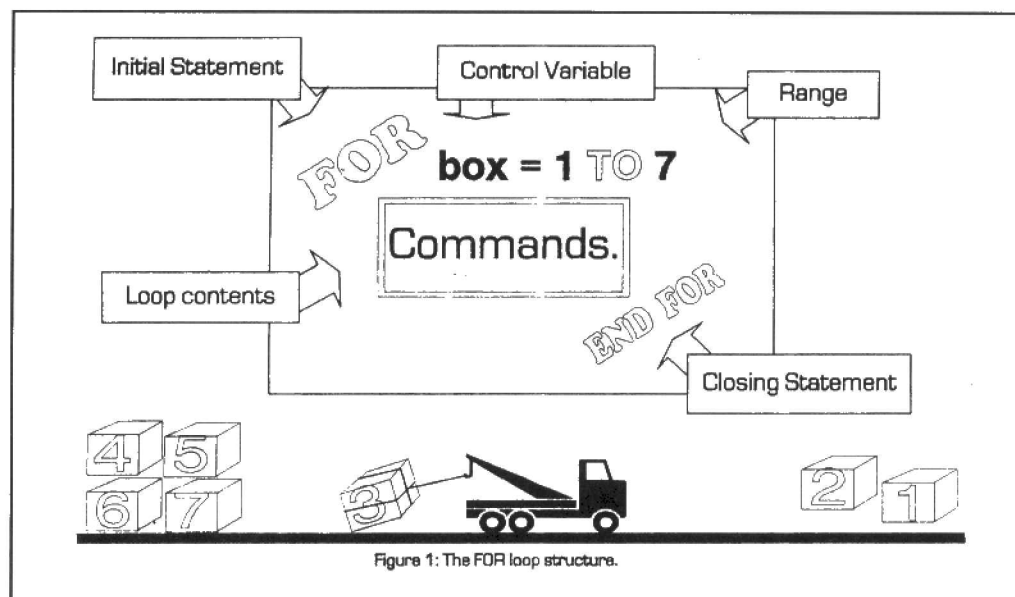
```
100 FOR N = 2 TO 12
110 PRINT N * 4
120 END FOR N
```

The variable in the FOR statement, N, is no different from other numeric variables, except that it is automatically incremented on each cycle of the FOR loop. Nevertheless, variables used in the control of loop structures have been given special names, such as 'control variable' and 'loop identifier'.

Note that the lines between a FOR statement and its END FOR successor are usually indented so that the structure of the program is clear from a glance at the listing.

Some program editors automatically increment lines in this way (*Archive* is a good case in point), but thanks to the QL's ultra-simple line editor programmers have to remember to do the job for themselves.

FOR N



FOR ... END FOR

The use of FOR...END FOR has radically improved the clarity of the program, but the earlier promise to reduce five lines to one has yet to be achieved. Alone among all Basic dialects, SuperBasic employs the concept of a 'short form' of its structures. This means that, provided that a structure is wholly contained on a single line, the final statement can be omitted.

The example programs in this series have so far contained one statement per line. To place many statements on a line it is necessary to separate them using a colon — the more English-like fullstop cannot be used to end statements because of its role as a decimal point. The FOR loop can therefore be shortened to:

```
100 FOR N = 2 TO 12: PRINT N * 4
```

SuperBasic makes the same assumption that we do when obeying the instruction 'For values of N between two and twelve...', ie that only whole numbers are intended. This is not always going to be true, so SuperBasic allows different increments to be specified. The numbers between four and 100 which are divisible exactly by four can be quickly listed using the line:

```
100 FOR N = 4 TO 100 STEP 4: PRINT N
```

STEP

The English equivalent would be 'Print every fourth number between four and 100'. The STEP number does not have to be a whole number, nor does it have to be positive. The following two examples count down from ten to one and count up from zero to one in small increments:

```
100 FOR N = 10 TO 1 STEP -1: PRINT N
200 FOR N = 0 TO 1 STEP 0.2: PRINT N
```

FOR loops can be embedded within other FOR loops. You might, for example, want to print every one of the multiplication tables from two to twelve using one program:

```
100 FOR number = 2 TO 12
110 FOR multiplier = 1 TO 12
120 PRINT number;" x "; multiplier;
130 PRINT "=" ; number * multiplier
140 END FOR multiplier
150 END FOR number
```

Note the way that incrementing the line indents highlights the structure of the program and makes it easy to see where each loop begins and ends. The important thing to note here is that the 'multiplier' loop is wholly contained within the 'number' loop. It should be obvious that if the *END FOR multiplier* line occurred after the *END FOR number* line then the program would produce nonsense. The concept of containment extends to the use of GOTOs inside FOR loops: they should not direct the interpreter to a point outside the bounds of the FOR and END FOR statements.

An interesting feature of this routine is that it prints 132 lines of information using just one more program line than our initial example. It is a simple demonstration of the way in which computers can be programmed to undertake repetitive tasks with very little effort. The program can be expanded to provide the first 100 multiplication tables, each one with multiples between 2 and 1000 if necessary, simply by changing the values in the FOR loops.

FOR loops are fine provided that the programmer knows in advance the range of values which will be used. There are many occasions when this will not be the case. The first resort is to change the absolute range values for variables.

Take a routine to print out all of the dominos from double blank to double six: the obvious loop structure will print out more dominos than there are in a set:


```

100 FOR left = 0 TO 6
110 FOR right = 0 TO 6
120 PRINT left; " : "; right; " "
130 END FOR right
140 PRINT
150 END FOR left

```

A quick check of the printout from this program reveals that dominos such as 3:5 are also included as 5:3. The only dominos not to be printed twice are the doubles. The listing needs to be altered to remove the duplicates by changing the inner loop:

```

100 FOR left = 0 TO 6
110 FOR right = 0 TO left
120 PRINT left; " : "; right; " ";
130 END FOR right
140 PRINT
150 END FOR left

```

The best way of understanding the exact effect of this seemingly minor change is to type in the listing and run it. Watch carefully for the punctuation in the PRINT statement on Line 120.

Programmers can also feel trapped by the regularity of the STEP feature of FOR loops. What happens if the values taken by the loop control variable are not equally spaced? For instance, what if the first six prime numbers had to be printed? SuperBasic, uniquely, permits a series of comma-separated values to be used instead of a range, which allows it to meet our needs perfectly:

```

100 FOR prime = 1, 2, 3, 5, 7, 11
110 PRINT prime
120 END FOR

```

The FOR...END FOR structure has many sophistications, some of which have yet to be covered here, designed to cope with specific circumstances. However, enough of this powerful structure's abilities have been covered here to cope with most needs.

There will be circumstances where a loop will cycle an unknown number of times, perhaps even permanently until the program is exited. This could be simulated using a FOR loop with some very large parameters, but SuperBasic provides a better way with another structure called the REPEAT loop.

Like the FOR loop, REPEAT loops have special statements at their beginnings and ends between which appear the program lines which are to be repeated. Here is a typical example of a REPEAT loop:

```

100 REPEAT loop
110 PRINT "Hello ";
120 END REPEAT loop

```

The characteristic capitalisation of the word 'REPEAT' is forced by SuperBasic. It indicates that lazy programmers can get away with typing only REP and leaving the Basic editor to do the rest. This feature only applies to the longer structure-related keywords, which makes one wonder why it was adopted at all.

Returning to the code in hand, the effect of putting the PRINT command between the REPEAT statements is that it is repeated forever, or at least until either the CTRL-SPACE key combination is pressed or the computer is reset. A short form of the REPEAT structure is available governed by the same rules used in short FOR loops. The above example could be rewritten as:

```

100 REPEAT loop: PRINT "Hello ";

```

This is the exact equivalent of the old chestnut typed countless times into shop display computers:

```

10 PRINT "Hello ";
20 GOTO 10

```

A more substantial example of a REPEAT loop is contained in the original *QL User Guide* (Dec 1984 reprint), but it is not a good example of programming. The routine plays a guessing game: a number is randomly thought of by the computer and a succession of inputs are requested until the player guesses the number. A neater version of the program is:

```

100 LET target = RND(9)
110 REPEAT gameloop
120 INPUT "Guess the digit.."; guess;
130 IF guess = target: EXIT gameloop
140 PRINT " is wrong"
150 END REPEAT gameloop
160 PRINT "Well done!"

```

REPEAT loops

EXIT

The vital point about the structure of this game is the inclusion of the EXIT command, otherwise the REPEAT loop would repeat forever. It is usual for EXIT commands to be contained within IF structures, otherwise they would be acted upon during the first cycle of the REPEAT loop, which rather takes away the point of it all. Note that, as in END REPEAT statement the loop's name must be included in EXIT statements.

Loops can contain as many EXIT statements as might be needed, although it is usually a good idea to stick to just one or two to avoid confusing yourself. EXIT statements also have a role to play in some FOR loops where, again, they must be followed by the loop identifier. The value of the EXIT command is that it tells the QL that the loop is finished with and can be forgotten. If a GOTO command was used instead, the QL would spend the rest of the program thinking it was still somewhere in a big loop.

The short form of the FOR and REPEAT loops can be borrowed by the IF structure. This provides consistency, meaning that programmers have less to remember when phrasing their commands, and it can ease readability by removing the THEN keyword. Here is a short IF clause:

```
IF x = 5: PRINT x
```

The compatibility between short forms should alert you to the possibility that there is a long form of the IF structure. Only a few Basic dialects allow multi-line IF structures, but SuperBasic is one of them:

```
100 IF account < 0
100 PRINT "Account in red!"
120 LET overdraft = 1
130 INK 2: PAPER 7
140 END IF
```

IF...ELSE

There is, of course, no point in having an EXIT clause in an IF statement, nor is there a structure control variable with which to label the EXIT, and so compatibility does not quite stretch this far.

A very useful addition to the IF clause is the ELSE keyword. Very often, two courses of action suggest themselves according to whether or not something is found to be true. In English we might say 'If it is dry I shall go shopping, otherwise I will read a book'. The computer's equivalent is very similar:

```
100 IF account > 0
110 INK 0
120 ELSE
130 INK 2
140 END IF
150 PRINT account
```

There are times when even two courses of action are not enough to cope with circumstances. It is possible to nest IF statements so that the program becomes a succession of IF..ELSE..IF..ELSE lines, but a much neater structure is available using the SELECT keyword. Using SELECT, a number of ranges can be tested for. Each range can have a set of actions associated with it.

NOTES TO LISTING 1

```
LINE 110: A random number is selected.
LINE 150: Each guess is counted...
LINE 160: The gap between the guess and the
          mystery number is calculated.
LINE 180: Every loop must have its EXIT!
LINE 200: Be careful with negative number
          ranges.
LINE 260: The result is printed.
```

LISTING 1

```
100 CLS
110 LET X = RND(100):
120 LET TRIES = 0
130 REPEAT GAME
140 INPUT "Your guess ... ?": GUESS
150 LET TRIES = TRIES + 1
160 LET DIFFERENCE = GUESS - X
170 SELECT ON DIFFERENCE
180   = 0 : EXIT GAME
190   = 1 TO 6 : PRINT "Just too high"
200   = -6 TO -1 : PRINT "Just too low"
210   = 7 TO 20 : PRINT "Too high"
220   = -20 TO -7 : PRINT "Too low"
230   = REMAINDER: PRINT "Miles out!"
240 END SELECT
250 END REPEAT GAME
260 PRINT "You won in ": TRIES: " guesses"
```

To demonstrate SELECT's value, let us expand the guessing game a little so that the users are given hints according to how close they get. See **Listing one** for the program code. The difference between each guess and the target number is calculated and then tested to see in what range it fits. An appropriate message is then printed. Notice the REMAINDER keyword, which must be the last range to be specified if it is used. Readers comparing the syntax with that appearing in the old User Guide will see that the program example is much more concise and easier to read.

The use of ranges in the SELECT structure allies it more with the FOR loop syntax than with that of IF statements. In fact, any ranges and lists which are permitted in FOR commands are also permitted in SELECT commands. This is only sensible because it means that both QLs and programmers need only remember one set of syntax rules.

The intelligent application of structures provides the skeletons for programs: the other commands are merely the flesh. With this section of the New User Guide your programming horizons have been greatly expanded. It may be surprising, therefore, that we have yet to reach the commands which make SuperBasic super...

Mention Archive to me, and I think of dusty files in a dimly-lit basement—maybe a stoop-shouldered, wire-frame-spectacled archivist totters among the shelves, dwindled into a hoop by years of poring over yellowed bits of paper. Of course, there is also the QL program of this name. For many — me among them — this original piece of free software has the same charm as a real-life archive, and is just as useful for everyday purposes.

There are those who use *Archive* as their standard database, and seem to get on with it. I gave it up as soon as possible, and have never regretted it. Instead, I turned to *FlashBack*, a user-friendly program with which I have had a relationship of growing fondness. Recently, *DATAdesign* has shown signs of coming between us — chiefly because it has the advantage (for me) of having been designed to operate under QRam or Qpac.

To begin with, let's be fair to Archive. This program is unbeatable in some areas. It has many features, but its main advantages, I would say, are:

1. The size of file is not limited by your computer's memory, because information is stored on whichever medium you use.
2. Your screen display is user-definable by using the Sedit procedure.
3. Archive offers a sophisticated programming language.
4. You can have more than one file open at once, to make links between them.

So Archive is in the heavyweight class, and is very flexible. Neither of the alternatives I have used can match its flexibility and range of facilities.

Archive is not without its problems, though. Of course, what seems to one user a problem, may not seem so to another, so this summary of the major drawbacks of Archive is another personal list:

1. Commands have to be typed in full — and correctly!
2. Archive is picky over things like spaces and trailing quotation-marks — and not always consistently so.
3. It is dreadfully slow.
4. I have always been terrified of failing to 'close' a file and corrupting it.

Compare either *FlashBack* or *DATAdesign* and you quickly see their advantages in simplicity of use. In *FlashBack*, two or three key-presses at most are enough to access the commands. In *DATAdesign*, a quick flick of the mouse is sufficient. Since both hold complete files in memory, you have to try quite hard to make a mess of your database — I have not managed it yet, and I rate high in the carelessness stakes. Both are phenomenally fast at searching a file, partly because they hold the file in memory and have no need to access a medium, but

ALTERNATIVES TO ARCHIVE

■ **FlashBack and DATAdesign are popular variations on the database theme. Michael Evans compares.**

largely also because both are very nifty examples of the programmer's art. There will be a couple of rough-and-ready benchmarks later.

So, what have *FlashBack* and *DATAdesign* got in common? Both are simple card-index systems. Both are memory-based. *FlashBack* in its Special Edition and *DATAdesign* need memory expansion. Neither of them offers much in the way of screen design — just simple re-sizing or re-positioning or alteration of default colours. Both can contain multi-line records in any field. Both cost money!

There are differences in concept. *DATAdesign* is menu-operated, and you navigate with a mouse, or with cursors and ENTER. You can also access commands by their (usually initial) letter-code.

You cannot use the QJump mouse with *FlashBack*, but navigation and commands are just as easy. There are three ways of accessing commands: CTRL/letter, F3 and letter, or from the commands list. *FlashBack* is not menu-operated; if you don't mind memorising the commands, you may prefer not to have the nuisance of menus constantly popping up is a nuisance.

You can have more than one *DATAdesign* present, if your computer's memory runs to it, and if you think there is any point. You can have more than one *FlashBack* too, and each can use two screens, which handle data in slightly different ways.

Now let's look at some specific features of these programs. They offer similar fa-

	FlashBack (*SE)	DATAdesign
File handling:	read write save as *selective write merge text file	load save
Data handling:	next back first last delete *index group. . . kill - kill create search. . . *replace view edit (fieldname)	next previous begin/first end/last delete sort n/a search view new field erase field truncate forget file
Output:	*xclude/include * (separate) (report) (generator) printer transfer text	print panel
Screen handling:	undo changes adjust window re-position	resize (standard) (QJump features)

ALTERNATIVES TO ARCHIVE

cilities, though they sometimes use different commands as shown in this table (the * means that the facility is available only in the Special Edition of FlashBack):

It is immediately apparent that FlashBack contains more features than DATAdesign. This does not necessarily mean that FlashBack is better! Let's look at the differences a little more closely.

It's initially disconcerting, for instance, that DATAdesign has no 'create record' command. In fact, the design of the program means that such a command is unnecessary: a file always contains a blank record, and as soon as you fill it up, another blank record is created to replace it. So to add a record, you just move to the last record in the file. This seems to me to give it an edge over FlashBack, especially because the quick way to create a record in FlashBack is CTRL/c — which means that if you are multitasking, attempting to create a new record this way will just take you on a review of your current tasks! Fortunately, of course, you can also use the F3/c combination to perform the same function. Here, then, we have to give top marks to DATAdesign.

The same attention to ease of use in DATAdesign is evident in the need for a Save As command. It's necessary because, if you Save, the program automatically updates the filename you last loaded in. If you want to change the filename, you use Save As.

Another file-handling facility in DATAdesign is Forget — which simply empties memory. Not, perhaps, an essential command. In FlashBack you can do the same job by reading in an empty file.

Against this, FlashBack has the ability to merge a text file — a facility I have yet to use — and, in the special edition, a selective Write option which is very useful. You can Group (Archive would say Select) a range of records by reference to some feature, say, for example, including 'Scotland' in the address, and save only those records as a separate file. Then you can use the Report Generator to print out all those addresses.

Although both programs have a command for ordering records — Sort in DATAdesign, and Index in FlashBack Special Edition — the operations are remarkably different. DATAdesign invites you to specify sorting on two levels, in normal or reverse order, as letters or as numbers. FlashBack SE offers the order 'ABCabc', 'AaBbCc', or 'aBcDeF' (case-independent), and may sort either alphabetically (the default) or numerically. Non-special FlashBack offers only a simple ordering procedure by means of the Group command. All three programs allow ordering on specific fields — ie, in a phone-book file, you could order the names, or (if you're better at remembering numbers than names) you could order the telephone-numbers.

One other important difference between DATAdesign and FlashBack SE data-han-

HELP F1	COMMANDS	create	look	open	close	COMMANDS F3
PROMPTS F2	delete	display	back	alter	find	ESCAPE ..
	first	insert	last	next	quit	ESC
	type command & press ENTER (F3 for more)					

```

Logical name : main
Library%    : C.A.D.1
Title%      : QL SCRIPTULA
Category%   : C.A.D.
Submitted%  : 28.02.86
Author%     : COGSWELL .E
Revised%    :
Text1%      : Computer aided design packages, one in Superbasic & a compiled ve
Text2%      : Instructions on SCRIP_DOC with a "Beginners Guide" by Brian Davie
Text3%      : STARTUP_DOC. Example data - SQUARES & HOUSEPLAN. Load basic versi
Text4%      : Lrun mdv1_scrip_bas & compiled version by Exec_w mdv1_scrip_cde
Text5%      :
Text6%      :
Text7%      :
Text8%      : LIB_SCRIPS/D_09
  
```

```

>look "libguide_dbf"
>display
>
  
```

Archive screen

dling facilities is the latter's Replace command, more often associated with word processing, but probably very useful even though I haven't yet needed to use it for my own limited purposes. Like some of the other facilities in FlashBack it gives you a feeling of security to have it there should you ever want it.

DATAdesign is very much a stand-alone program — an executable file dependent only on the QJump pointer environment, which enthusiasts are likely to have installed more or less permanently anyway. The single program does everything the

package offers for its files, including printing. The printing procedure is both simple and complicated. A panel opens up on screen, into which you type the details of the output you want.

So far so good. But the mechanism used to determine layout and spacing is ingenious enough to make accurate typing, or at least careful checking, quite important. A number of characters have to be typed into a 'form string' to instruct the computer — 'f' for a form feed, 'n' for whether fieldnames are printed, and so on. Whenever you want to print a file, you have to

```

C.A.D.1  QAL SCRIPTULA  C.A.D.  028.02.86  COGSWELL .E  Q Computer aided des
C.A.D.1
QAL SCRIPTULA
C.A.D.
028.02.86
COGSWELL .E
Q
QComputer aided design packages, one in Superbasic & a compiled version.
QInstructions on SCRIP_DOC with a "Beginners Guide" by Brian Davies on
QSTARTUP_DOC. Example data - SQUARES & HOUSEPLAN. Load basic version by -
QLrun mdv1_scrip_bas & compiled version by Exec_w mdv1_scrip_cde
Q
Q
Q
QLIB_SCRIPS/D_09
  
```

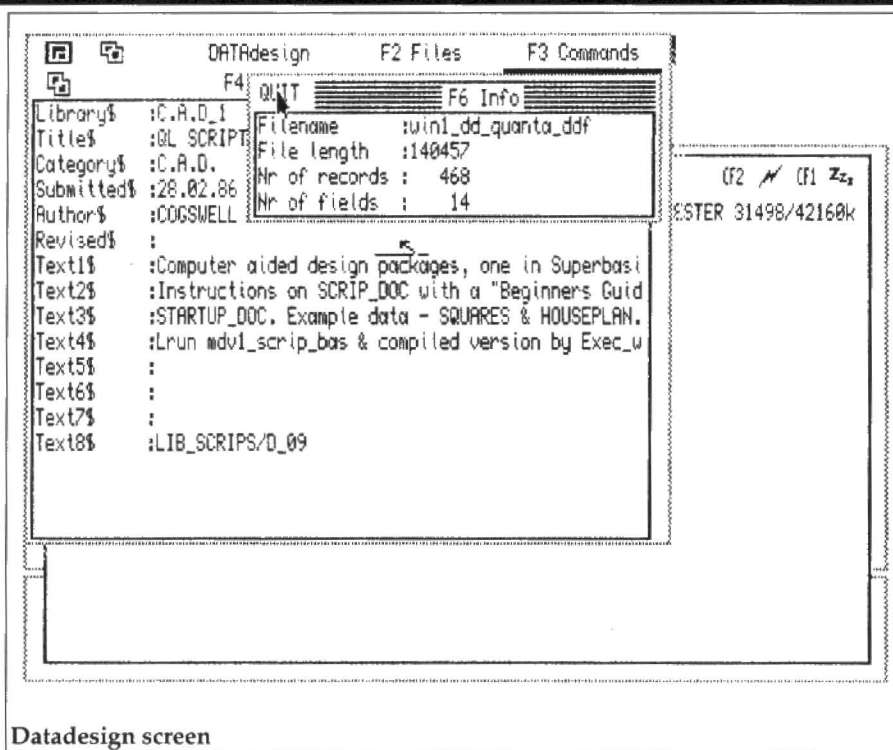
```

01 Library
010000
  
```

```

00005  00468
- I - win1_FB_quanta_dba
  
```

Flashback screen



Datadesign screen

type in the form string again. At the same time, you can determine which records you wish to print by setting up a comparison string against which each record in the file is measured.

Another important point about the printing mechanism is that, if it comes across a problem, it just aborts. No message! Would it be so hard to introduce some simple indicator of roughly what has gone wrong?

With FlashBack Special Edition you get a package including a separate Report Generator, and sample templates which you can use or alter, and read into the computer to determine, for example, the layout of an address label. Although setting up the system initially is quite complicated, once you have completed it the job is done for good, or until you want to alter it. (In non-special FlashBack there is no Report Generator — just a simple printout command which prints whatever records are currently selected by the Group command).

Here we have two radically different approaches to solving a single problem. Each has its advantages.

Both packages have a separate 'import' program for tailoring Archive files —

FlashBack SE can use *Abacus* files, too. The FlashBack system works fine. So does the DATAdesign system, but much more slowly. I imported the Psion Gazet .dbf file into both programs: FlashBack did it in two minutes; the SuperBasic DATAdesign import program took 15 minutes! The DATAdesign screen gives you no sign, while it is organising its file, that anything at all is happening, so it's important not to press the reset button on the assumption that your QL has gone off on holiday again.

Of course, such translation is called for only once for each file, so you may feel that the expenditure of time is not very significant. Computer users waste so much of it anyway, fighting with various features of their system, don't they?

I took some simple measurements with a stopwatch on Archive, FlashBack SE, and DATAdesign. In each case I used the Quanta Library Guide. I did not use the fancy Quanta screen with Archive, which I feel slows things up a little further, although it improves ease of use. I used hard disk with each program, but in the case of Archive, which is more medium-dependent for its performance, I have also given times using floppy disk. I'm afraid I did

not have the patience to attempt micro-drives! These are the results, with all times in seconds:

The quickest runner, clearly, is FlashBack. Non-special FlashBack is actually fractionally faster — presumably because of its simpler configuration. Although the figures for DATAdesign appear quite a lot poorer, both programs are so fast that the difference doesn't really matter for practical purposes. Certainly, I don't feel that the difference in speed is important enough to play any part in determining which program to buy. Other factors seem to me much more significant, and both easily outclass Archive.

Much, but not everything, depends on whether you are a user of the QJump pointer system. If you are, DATAdesign will use it effectively, and you are likely to take to the program like the proverbial duck.

FlashBack, on the other hand, is not designed to work with QRam or Qpac. Nevertheless, you can use it alongside the pointer environment, and you can multitask it. I do. The important thing to remember is that you must always go to SuperBasic before accessing FlashBack. With that proviso, you can have as many jobs going as your computer's memory can cope with. You may well need to make some adjustments to the program, or to its basic loader, to operate it successfully with QJump products. I had to alter cursor speed in FlashBack SE to use it with Minerva and Qpac; and I had to amend the Report Generator boot program to read filenames properly with the same combination of hardware and software. I found QView (Minerva distributors) and Sector Software, who distribute the program, very helpful in this respect; no doubt Dilwyn Jones Computing, who are now also selling FlashBack, will be equally prepared to advise on modifications if required.

Non-special FlashBack is around half the price of DATAdesign, and represents good value for money. But it does not, in my view, offer significant advantages over DATAdesign, which, on the other hand, does have a great deal going for it — ease of use, the pointer environment, and a clever (and superior) built-in printing system.

It's different when you come to FlashBack SE. This is a really powerful program, which does everything I want of a database, and a bit more. The fact that the manuals for both FlashBack and the Report Generator are FlashBack files, which can be navigated through with cursors and function keys, shows what the program is capable of. At roughly the same price as DATAdesign, it offers a wider range of facilities at the cost of considerably more complexity.

This isn't a criticism of DATAdesign, which achieves its aims perfectly — it just doesn't aim at doing all that FlashBack does. In the end, any decision between the two comes down, as so often, to individual preference.

	Archive (Flp Win)	FlashBack SE (Win)	DATAdesign (Win)
Access the file and display first record	15 12	4	19
Find 'Z88' from first record with the file in its raw state	8 6	1	1
Sort the file by titles in ascending order	82 39	6	22
Find 'Z88' from first record in file sorted by titles	46 34	4	12

Desires & Devices

Many QL owners have now moved on from microdrives to disks of one sort or another and this presents a problem to the programmer, who should try hard to cater for the user's particular set up, whether two mdvs, two flps, one fdk plus two mdvs, or whatever.

The extent to which a program is capable of taking into account these different configurations will be a major element in its success. But how can a program know its user's configuration?

The object of this article is to describe a method of accomplishing this from SuperBasic. There are ways of doing the job with machine code, but they are not dealt with here.

Storage

Let us assume that you are writing a SuperBasic program which requires two storage devices: one for programs and basic data files and another for the files that the user will generate by using your program. The devices may be mdvs, flps, fdks or any two from those three.

You have no way of knowing which configuration will be used. You cannot assume that because your program was ordered and supplied on, say,

I QL systems are getting more complex, Brian Storey suggest ways of allowing configuration simply from SuperBasic – ways which work, and ways which don't.

a floppy disk that the user will not wish to store its products on a microdrive cartridge. After all, not everyone has two disk drives.

Let us also assume that your program is governed by a BOOT program which will be called, and RUN in the traditional manner by pressing F1 or F2 from the Sinclair copy-right screen.

To get to know your user's configuration, you could ask by means of an INPUT statement for the mnemonic (FLP1, MDV1, etc.) of the device which carries the supplied medium and for that of the device which is going to carry the program's outputs (FLP2, MDV2, etc.) You can then use the two strings (prog\$ and file\$, say) as points of reference for the main program's functions.

Plausible, but it won't work!

The stumbling block is that when the BOOT program has accomplished its main task of LRUNning your main program, the precious data contained in your carefully constructed prog\$ and file\$ will no longer be there. Those strings will have been CLEARED by the LRUN command. They will no longer exist.

MRUN?

OK, why not MRUN the main program instead? That will work – no question.

MRUN instead of LRUN in your BOOT will not destroy prog\$ and file\$, so they will be available to your main program.

MRUN, however, carries its own constraints. When your main program is MERGED and RUN by use of MRUN, your BOOT program will remain in ram as a fairly useless preamble to your main program, wasting storage space for what may be a lengthy main program. Besides, you may wish the main program to call and RUN other programs. You would have to MRUN those as well in order to preserve prog\$ and file\$. And you would have to be very careful with your line-numbering to ensure that extraneous statements in the main program did not foul up your subsidiary program after the MERGE. Such difficulties are manageable, but they are a bit of a bore and prone to programmer error.

There is a fairly simple solution to all of this. There may be many more, but this one may appeal to you.

The nub of the problem is to preserve prog\$ and file\$ *after* the BOOT program LRUNs your main program. In the BOOT, you must ask the user to state his mnemonics for Drive 1 (programs) and Drive 2 (program generated files). Test the feasibility of those data to the extent that you judge them to be crash-proof, and then (the problem) preserve them in such a manner that they will not be destroyed by further operations.

Try doing it like this . . .

There are, tucked away in the bowels of the QL, some bytes known as System Variables. They are accessible from SuperBasic (PEEK) and even capable of amendment (POKE), although altering their values is not usually recommended because Qdos uses them to run the QL system.

Reserved

However, there are some of these 'reserved bytes' that are NOT used by Qdos and, moreover, they are not corrupted by LRUN. By default, they are set to zero. So let us store this essential mnemonic data in those unused bytes, and our problem is solved!

The spare System Variables that I use for this purpose are 10 bytes numbered 164068 to 164077, inclusive. The first five are used to store the contents of prog\$ and the rest to do the same for file\$.

Please examine Listing one, which is a typical BOOT program. The numbers in the following commentary refer to line numbers in the listing.

110 calls the PROCEDURE 'get_devices' which begin at 160. The code inside loop a (170 to 210) asks the user for the mnemonic (mdv1, flp1, etc.) of the PROGRAM file and then checks its feasibility in the PROCEDURE 'check' (370 to 430) which uses the received parameter prog\$ as its x\$.

Flat to 1

If the string is empty (390) or its penultimate byte is not numeric in range 1 to 8 (410) — you can't have more than eight mdvs or eight disc drives on a QL — or its length is not equal to five bytes (420) it sets a flag to 1 to indicate non-feasibility and RETURNS to the PROCEDURE 'get_devices'. If that PROCEDURE finds the flag set to other than zero, it REPEATS loop a which has the effect of asking the user for the same information again. You could improve the presentation by arranging for a disapproving BEEP if the data is bad and even PRINT out a message as to what is wrong if you want to be really helpful.

The 'check' PROCEDURE also

adds an underscore symbol to prog\$ (390) if the user has not typed in one of the pesky things.

Loop b (220 to 260) of the PROCEDURE 'get_devices' does all of this again for the mnemonic of the 'files' device.

So now you have the required device mnemonics in prog\$ (say 'mdv1_') and file\$ (say 'mdv2_'). It is time to preserve them.

At 120 the BOOT program calls the PROCEDURE

'record_device' (290 to 430) which POKES the CODEs of successive bytes of prog\$ into the spare System Variables, 164068 to 164072 (300 to 330). Then it takes similar action, by making a second call to 'record_device' (with different parameters), for file\$.

Now we're in business. It only remains to LRUN your main program (140) which will be able to identify its devices from the data recorded in locations 164068 to 164077. That

data will persist until you reset your QL. It will survive CLEAR and even NEW.

Your main program will, however, need to convert the device data from its numeric form into string form. This is where Listing two comes into play.

You will see that 1000 sets up two 'null' strings prog\$ and file\$, and that 1010 to 1020 loads the character equivalents of the CODEs from the System Variables locations into those strings. This coding must appear in your main program before you attempt to access any device.

Flexible

You can now easily and reliably issue statements such as those in 1040 to 1060. You have made your program flexible enough to cope with all the devices that your user may employ on his system.

All of this relies upon those spare system variables being left unaffected by normal operations. If they get corrupted, your program will crash with a 'not found' error from Qdos — and perhaps other unhappy things will occur.

However, it all works in a pristine machine. Note that it may not work in the presence of various proprietary software such as toolkits. But then, you wouldn't write a program for general use that depended on toolkits, would you? Better remember to mention that in the Manual, though . . .

Reading

In reading this, is may have occurred to you that these mnemonic data could be tucked away in a short file on cartridge or disk, where they could be accessed by your main program without having to ask the user to type them in every time he/she RUNS your program.

Yes, I thought about that. I even coded it!

And then — you are ahead of me? — I realised that my program would not know, unless it asked, where to READ the file from!

What fools we mortals be!

LISTING 1

```

100 CLEAR:CLS:CL#0
110 get_devices
120 record_device 164068, 164072, prog$
130 record_device 164073, 164077, file$
140 LRUN prog$ & "Main_prog"
150 REMark *****
160 DEFine PROCEDURE get_devices
170 REPEAT loopa
180 INPUT "ENTER mnemonic of device which carried your
PROGRAM":prog$
190 check prog$
200 IF flag = 0: EXIT loop a
210 END REPEAT loop a
220 REPEAT loop b
230 INPUT "ENTER same for device which will carry your
FILES":file$
240 check file$
250 IF flag = 0: EXIT loop b
260 END REPEAT loop b
270 END DEFine get_devices
280 REMark *****
290 DEFine PROCEDURE record_device (a,b,x$)
300 LET x = 1
310 FOR n = a TO b
320 POKE n, CODE (x$(x))
330 LET x = x+1
340 END FOR n
350 END DEFine record_device
360 REMark *****
370 DEFine PROCEDURE check (y$)
380 LET flag = 0
390 IF y$="" THEN LET flag=1:RETURN
400 IF y$(LEN(y$)) <> " " THEN LET y$ = y$ & " "
410 IF NOT y$(LEN(y$) - 1) INSTR "12345678" THEN LET flag
= 1:RETURN
420 IF LEN (y$) <> 5 THEN LET flag = 1:RETURN
430 END DEFine check
440 REMark *****

```

LISTING 2

```

1000 LET prog$="" :LET file$=""
1010 FOR n = 164068 TO 164072:LET prog$ = prog$ &
CHR$(PEEK(n))
1020 FOR n = 164073 TO 164077:LET file$ = file$ &
CHR$(PEEK(n))
1030 REMark *****
1040 DELETE prog$ & "temp_prog"
1050 OPEN NEW#3, file$ & "file_1"
1060 REMark *****

```


DIY TOOLKIT



Simon Goodwin finds existing file display commands lacking, and presents the fast, friendly DIY alternative – the MORE command.

I am always eager to hear of new commands that QL users wish to use on their machines. This project was suggested by local *Quanta* group Secretary Stuart Jones, who asked for a command to display files as pages of text in a window, allowing movement back and forth.

'What, MORE?', I said, relishing the chance to write some dialogue, and thinking of the program of that name in Bell *Unix* systems. MORE lets you look through a file by pages or lines, showing your position in the file as you go along. Unix implementations often show the percentage of the file that has yet to be displayed.

There is little to choose between fractions and percentages, and sometimes exact information is useful; the DIY command echoes the file name, size and current position, in bytes. You can enter a new position at any time, moving directly to a particular character of the file.

ALT up-arrow and ALT down-arrow move back and forth a page at a time. I chose these key-strokes to be compatible with the PgUp and PgDn keys on PC-style keyboards, available for the QL and fitted on my Thor XVI. Down-arrow scrolls the screen, a line at a time. It is much faster to

move in pages.

Of course, you could do all this, and far more, with a separate program like *FEDIR*, *Spy* or *The Editor*, but these are quite large tasks; they need the file in memory, plus indices, and may well use buffers outside the task as they run.

QL MORE is a pages display command for SuperBasic, instantly available without disturbing the main memory, using the default or designated sub-directories and windows. As the whole thing is written in machine-code, it fits in just 1K of memory, and can be shared between any number of SuperBasic tasks.

Copy to screen

If MORE is to be useful, it must be superior to the customary commands that display files: VIEW and COPY. Discussions with other QL users confirm that I am not alone in using COPY TO SCR to scan tasks and Psion *Quill* documents.

If you just want to know what is inside, it is much less disruptive to copy a DOC to SCR than it is to load *Quill* or *Xchange*. COPY can reveal the extensions in code files; you could even cheat at adventure games by looking through the code for

commands and messages!

The COPY command is built into every QL, with parameter handling extended by Toolkit 2. It opens two channels and copies bytes from the first to the second until an error occurs, when it closes both channels.

The Toolkit extensions make the first parameter default to the DATA_USE sub-directory. SPL_USE gives a default for the second parameter. SPL_USE 'SCR_512X256a0x0' makes the default COPY device a big window, so COPY BOOT display the BOOT file on the full screen. It also directs the SPL command to SCR, rather than the usual SER or PAR. You need *QL Toolkit* or *Toolkit 2* to use these defaults.

COPY FILE, SCR opens a window in the smallest character size for the MODE that you are using, and scrolls the bytes of FILE through that window. The colours, like the CSIZE, are fixed: green ink on black paper. You can over-ride the default window size by adding parameters after SCR; recent Argos systems let you specify a border at the same time.

Assuming FILE is too long to fit the window, COPY scrolls text continuously unless you press CTRL-F5 to pause all display output, disrupting other tasks that want to use the screen. There is no way to move backwards in the file; once text has scrolled away, it has gone and you cannot see it again unless you re-start the COPY from the start.

* QL WORLD DIY TOOLKIT - MORE keyword 1.1 by Simon N Goodwin, May-June 1991

```
#
buf      equ      a4          Replace BUF with A4 if you lack EQU
output_id equ      d5          Channel ID of the text output window
prompt_id equ      d7          Channel ID of the prompt window (#0)
#
```

* Names for variables in Buffer, addressed with NAME(BUF,A6,L)

```
#
width    equ      0           WIDTH of output window, in characters
depth    equ      width+2     DEPTH of output window, in characters
old_top  equ      depth+2     File point at top of previous page
column   equ      old_top+4    Loading COLUMN & LINE corrupts OLD_TOP
line     equ      column+2     with the width & height of #0
```

```
maxtext  equ      line+2      MIN( BUFF SIZE-TEXTBUF, DEPTH*WIDTH )
textleft equ      maxtext+2   Number of bytes unscanned in buffer
textptr  equ      textleft+2   A6 offset to first unscanned byte
digits   equ      textptr+4   Buffer for 10 numeric digits (I hope)
top_left equ      digits+10    File point at top left of next page
filename equ      top_left+4   Space shared by file name, header etc.
length   equ      filename     LONG File length & start of file header
filetype equ      length+5     Offset to File type byte in file header
dataspace equ      length+6    Offset to LONG data space in header
headname equ      length+14    Offset to name in file header
textbuf  equ      length+4     From here on is file text buffer space
#
start    lea.l      define,a1   Point at the details
         move.w     $110%w,a2   Use the BP.INIT vector
         jmp        (a2)
#
```

```

* MORE [ # CHANNEL# , ] FILE_NAME
*
more    moveq    #1,d1      Assume channel #1
        lea.l    8(a3),a4
        cmpa.l   a4,a5      Just one parameter?
        bmi      bad_param  Reject unless there is one!
        beq.s    channel
        exg      a4,a5      Isolate the first parameter
        move.w   $112(w,a2  CA.GTSTR - get an integer
        jsr      (a2)
        bne      bad_exit
        move.l    a5,a3      Start of next parameter details
        move.l    a4,a5      End of supplied parameter info
        move.w   0(a1,a6.l),d1 Grab the channel number
        beq.s    bad_param  Alas, #0 is already needed
channel  move.l    40(a6),d2  Find the channel table
        move.l    0(a6,d2.l),d7 D7 is ID of channel #0
        mulu     #40,d1      Find offset in table
        add.l     d2,d1      Add base to offset
        cmp.l     52(a6),d1  Check not beyond end
        bge.s    what_chan
        move.l    0(a6,d1.l),a0 ID for output window
        move.l    (a6),buf    Find SuperBasic's fixed BUFFER offset
        move.l    buf,a1      Optimise code if WIDTH=0 (expected)
*        lea.l    width(buf),a1 Generate this if WIDTH is changed!
        moveq     #-1,d3      Wait forever if necessary
        bsr      window_spec  Read DEPTH & WIDTH of main window
        bne.s    bad_exit     Windows only! Reject imposters
        move.l    a0,output_id Store ID for use later
        move.l    prompt_id,a0 Get busy in screen window #0
        moveq     #14,d0      SD.CORE; turn on the cursor in #0
        bsr      call_qdos
        bne.s    bad_exit     Ahem, something is terribly wrong!
        lea.l     column-4(buf),a1 Four words; MORE uses 3rd & 4th
        bsr      window_spec  Read COLUMN & LINE for prompt window
*
* Get the file name parameter as a string or a name
*
get_name  tst.b     0(a3,a6.l) Is the value set?
        bne.s    value_set    If so, use GTSTR to read it
        movea.l   24(a6),a0    A0 -> SuperBasic Name Table
        move.w     2(a3,a6.l),d0 D0 is index of parameter name
        lsl.w      #3,d0       Scale for eight byte entries
        adda.w     d0,a0        Implicitly extends D0 to match A0
        move.w     2(a0,a6.l),d0 D0 is offset of text in Name List
        ext.l      d0           Data regs. need explicit extension
        add.l      32(a6),d0    Add Name List base offset
        move.l     d0,a2        Save total offset for later
        lea.l      filename(buf),a3
        move.l     a3,a0        Set name offset for OPEN later
        moveq     #0,d1         Clear high bytes of D1
        move.b     0(a6,d0.l),d1 D1 is length of name (1..255)
        clrb       0(a3,a6.l)  Clear high byte of buffer word length
copy_name  move.b     0(a6,d0.l),l(a3,a6.l)
        addq.l     #1,a3        Advance through buffer
        addq.l     #1,d0        Advance through Name List
        dbra       d1,copy_name Copy D1+1 bytes including length
        bra.s      name_ready
*
what_chan  moveq     #-6,d0      CHANNEL NOT OPEN error
bad_exit   rts
bad_param  moveq     #-15,d0     BAD PARAMETER error
        rts
*
* OPEN subroutine, tries to OPEN_IN the name at A0
*
try_open  moveq     #1,d0        Trap key for IO.OPEN
        moveq     #-1,d1        This task will own the channel
        moveq     #1,d3        Try to OPEN_IN
        trap      #4            Remember A6
        trap      #2
        tst.l     d0            Set Z flag if it worked
        rts
*
value_set  move.w   $116(w,a2  CA.GTSTR - get a string
        jsr      (a2)
        bne.s    bad_exit
        subq.w    #1,d3        Only one more parameter, please
        bne.s    bad_param
        lea.l     1(a1),a2      Save offset of length byte
        move.l     a1,a0        Find the string start
name_ready  cmp.w     #120-33,filename-2,0(a0,a6.l) Allow default & data
        bpl.s    bad_param     Name too long, reject it
        bsr.s    try_open
        beq.s    Hurrah, it worked! Avoid next effort
* Search out the DATAD$ default device prefix among the system variables
*
        move.l     d0,d3        Save the error code for later
        moveq     #0,d0        MT.INF trap key
        trap      #1           Find the System Variables
        move.l     176(a0),d0    D0=0 or points to the DATAD string
        beq.s     fail_d3       No default so return report code in D3
        move.l     d0,a0
        move.w     (a0)+,d0      Pick up the string length
        bne.s     default_ok     If default is null, it won't help
fail_d3    move.l     d3,d0      Return error code from OPEN
        rts
*
* Copy default to the buffer, append the parameter, and re-try OPEN
*
default_ok  lea.l     filename(buf),a3
        move.w     d0,d1        Save length for total
        subq.w     #1,d0
        deft_copy  move.b     (a0)+,2(a3,a6.l)
        addq.l     #1,a3
        dbra       d0,deft_copy
        move.b     0(a2,a6.l),d0 Recall the length byte
        ext.w      d0
        add.w      d0,d1        Total length of the composite string
        subq.w     #1,d0
        slow_copy  move.b     1(a2,a6.l),2(a3,a6.l)
        addq.l     #1,a3
        addq.l     #1,a2
        dbra       d0,slow_copy
        move.w     d1,filename(buf),a3
        lea.l     filename(buf),a0
        bsr.s     try_open      Try to open file; name is at (A0,A6)
        bne.s     bad_exit     Had enough? Give up and complain!
*
* File is open, Channel ID is in A0 - read header & write prompts to #0
*
opened     moveq     #64,d2      Buffer size
        lea.l     length(buf),a1 Read the file header
        moveq     #-1,d3        Infinite timeout
        moveq     #71,d0        FS.HEADER
        bsr      do_trap4       Read file header
        move.w     d1,d6        Save header length for later tests
        exg      prompt_id,a0   Swap from file to prompt channel ID
        bsr      tidy_wind      Clear space for file position
        cmp.w     #64,d6        Was it a serial header?
        beq.s     full_head     Show file header details
        st        length(buf,a6.l) Mark the length as 'enormous'
        bra.s     find_room     Skip the other messages
full_head  lea.l     spacer,a1
        bsr      print_abs      Separate current position from size
        move.l     length(buf,a6.l),d1
        bsr      print_long     Show total file size in bytes
        cmpi.b     #1,filetype(buf,a6.l) Check A1 is still OK
        bne.s     not_task
        lea.l     lbracket,a1   Bracket data space
        bsr      print_abs
        move.l     dataspace(buf,a6.l),d1
        bsr      print_long     Show file data space in bytes
        lea.l     rbracket,a1   ...in brackets
        bsr      print_abs
        not_task  lea.l     spacer2,a1 Yet more gratuitous ornamentation
        bsr      print_abs
        move.w     headname(buf,a6.l),d2 Pick up the file name length
        lea.l     headname+2(buf),a1 Point at the text
        trap      #4            Text address is A6 relative
        bsr      print_str      Print file name from header
        move.l     8(a6),d0      Find end of buffer area
        sub.l     buf,d0         Subtract start to get size
        moveq     #textbuf,d1   Avoid slow LONG operand fetch
        sub.l     d1,d0         Leave room for buffer variables
        move.w     width(buf,a6.l),d1
        mulu     depth(buf,a6.l),d1
        cmp.l     d1,d0         Is the window bigger than the buffer?
        bls.s     screenfuls    Expects <= 32K characters/screen!
        move.l     d1,d0        Use all the buffer space we have
        screenfuls  move.w     d0,maxtext(buf,a6.l)
        clr.l     top_left(buf,a6.l)
        none_left  clr.w     textleft(buf,a6.l) Invalidate buffered text
        tas       old_top(buf,a6.l) Invalidate 'last page'
        exg      a0,prompt_id   That's enough prompting for now

```



```

100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
150 CLS: RESTORE : READ space: start=RESPR(space)
160 PRINT "Loading Hex..." : HEX_LOAD start
170 INPUT "Save to file...":f$
180 SBYTES f$,start,byte : STOP
190 :
200 DEFine FuNction DECIMAL(x)
210 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
220 END DEFine DECIMAL
230 :
240 DEFine PROCedure HEX_LOAD(start)
290 byte = 0 : checksum = 0
300 REPEAT load_hex_digits
310 READ h$
320 IF h$="*" : EXIT load_hex_digits
330 IF LEN(h$) MOD 2
340 PRINT"Odd number of hex digits in: "h$
350 STOP
360 END IF
370 FOR b = 1 TO LEN(h$) STEP 2
380 hb = DECIMAL(b) : lb = DECIMAL(b+1)
390 IF hb<0 OR hb>15 OR lb<0 OR lb>15
400 PRINT"illegal hex digit in: "h$ : S
420 END IF
430 POKE start+byte,16*hb+lb
440 checksum = checksum + 16*hb + lb
450 byte = byte + 1
460 END FOR b
470 END REPEAT load_hex_digits
480 READ check
490 IF check <> checksum
500 PRINT"Checksum incorrect. Recheck data."
520 END IF
530 PRINT"Checksum correct. data entered at: "start
560 END DEFine HEX_LOAD
570 :
580 REMark Space requirements for the machine code
590 DATA 1022
600 :
610 REMark Machine code data
620 DATA "43FA03EC34780110","4ED2720149EB0008"
630 DATA "BBCC6B0000946716","CB4C347801124E92"
640 DATA "660000084264D2A4C","3231E800677A242E"
650 DATA "00302E362800C2FC","0028D282B2AE0034"
660 DATA "6C62207618002856","224C76FF61000330"
670 DATA "66542A082047700E","61000328664843EC"
680 DATA "00046100031A4A33","E800664E206E0018"
690 DATA "3033E802E748D0C0","3030E80248C0D0AE"
700 DATA "0020244047EC0022","204B720012360800"
710 DATA "4233E80017B60800","E801528B528051C9"

```

```

710 DATA "4233E80017B60800","E801528B528051C9"
720 DATA "FFF4602870FA4E75","70F14E75700172FF"
730 DATA "76014E444E42A480","4E75347801164E92"
740 DATA "66E453436E245E9","000120490C70003B"
750 DATA "E8006AD461D6674C","260070004E412028"

760 DATA "00B0670620403018","660420034E7547EC"
770 DATA "0022320053401798","E802528B51C8FF8"
780 DATA "1032E8004880D240","534017B2E801E802"
790 DATA "528B528A51C8FF4","3981E82241EC0022"
800 DATA "618A6682744043EC","002276FF70476100"
810 DATA "02503C01CF886100","0238BC7C00406706"
820 DATA "50F4E822604643FA","02926100022A2234"
830 DATA "E822610001D40C34","0001E827661843FA"
840 DATA "0280610002122234","E828610001BC43FA"
850 DATA "02746100020243FA","0270610001FA3434"
860 DATA "E83043EC00324E44","610001EE202E0008"
870 DATA "908C722690813234","E800C2F4E802B081"
880 DATA "630220013980E80C","42B4E81E4274E80E"
890 DATA "4AF4E804CF88CB88","70204E43CB883834"
900 DATA "E8023C34E8003234","E80E67482274E810"
910 DATA "700A53416B36B031","E800528967125346"
920 DATA "62F04A41670AB031","E800660452895341"
930 DATA "3C34E800534462DA","240994B4E8109574"
940 DATA "E80E610001902989","E81060343434E80E"
950 DATA "610001823434E80C","43EC00262989E810"
960 DATA "70036100015C6706","0C0000F666083981"
970 DATA "E80E670C60962400","6100003220024E75"
980 DATA "720070434E433434","E80E48C292822601"
990 DATA "CF88610001342204","610000CE70014E43"
1000 DATA "4A806B06B23C001B","6612CF8870024E42"
1010 DATA "2047700F4E43720A","600000B8CF88B23C"
1020 DATA "00D8660CB8B4E822","671E78016000FF34"
1030 DATA "B23C00D9662629B4","E81EE8042984E81E"
1040 DATA "B8B4E8226600FF10","72002961E81E7042"
1050 DATA "4E43CF88610000BA","6000FEF2B23C00D1"
1060 DATA "66284AB4E81E6610","2234E82204810000"
1070 DATA "02006AD6CF886084","2234E8046AC00484"
1080 DATA "000008006BC22204","60C0B23C000A66E4"
1090 DATA "CF88617C81000092","740A43EC00147002"
1100 DATA "617E660E53416F0A","43EC001461000092"
1110 DATA "67042234E81E76FF","280170144E43700E"
1120 DATA "4E43CF8822046082","47EC00144A816606"
1130 DATA "72307005604C7409","7000484167243001"
1140 DATA "80FC000A48403200","484182FC000A3001"
1150 DATA "4841C34006000030","1780E809538B51CA"
1160 DATA "FFD843F420154442","064200094E446006"
1170 DATA "611643FA0005A3419","70074E434E75700B"
1180 DATA "4E444E434A804E75","3434E80A3234E808"
1190 DATA "701060EECB882274","E81061D0CB884E75"
1200 DATA "3401720076001631","E8005289B63C0039"
1210 DATA "6218040300306512","D2812001E581D260"
1220 DATA "D283534266E07000","4E7570EF4E75000A"
1230 DATA "2020202020202020","20200004206F6620"
1240 DATA "0002207B00017D00","000420696E200001"
1250 DATA "FC1A044D4F524500","000000000000"
1260 DATA "","*",87479

```

COPY has other weaknesses; it does not reveal what proportion of the file has been viewed, or the size of the file. It does not clear its window, so copied text can get muddled with other windows, or output from previous COPY commands.

COPY is a general-purpose command, designed to work with all devices, so it is not ideal for looking at files. However it does get used for that purpose, because it is always available and gets results quickly.

Tony Tebby's VIEW command surfaced in Sinclair's QL Toolkit, and later in Care Electronics' SuperToolkit2 and many disk systems. The manual says 'VIEW is a procedure intended to allow a file to be examined in a window'; in fact it is a way of reading Ascii files of short lines.

VIEW takes one or two parameters — a file name or string, and an optional channel number. By default output is to channel #1, but you can specify a device or any other open channel instead. VIEW #2, TEST displays the file TEST in the listing window.

VIEW is superior to COPY in that it can

use existing SuperBasic windows, defaulting to #1, like most display commands. Long lines do not wrap onto several display lines; instead VIEW truncates each file line to fit the window, showing only the start of long lines.

VIEW pauses each time the window has been filled with new 'page' of text. You do not have to press CTRL-F5 as the program does it for you, locking the display till a key is pressed. Each page appears a line at a time from the bottom. If the window has many lines this scrolling limits

No markers

the rate of text output.

VIEW works well with narrow Ascii files, like most assembler source and SuperBasic listings, but it is hopeless with tasks, binary data and Quill DOC files, as it tries to read a line at a time; such files do not have 'line end' markers!

VIEW is only suitable for files of lines, and you need a wide window to see all of each line. As with COPY, there is no way to move backwards, or to determine where

you are in the file, and the CTRL-F5 pause control disturbs other tasks needlessly. MORE fixes all these problems, and that's not all.

The simplest form of the command is MORE THING, where THING is the name of a file in the data sub-directory. The file may be identified by a string, string expression, or unset name. The default window is #1 but you can over-ride that with an explicit channel number. The output channel must be a window.

The parameters of MORE are like those of VIEW, although the output channel is expected to be open. VIEW & SCR, THING copies file THING from the default data directory to a temporary SCR channel. To do this with MORE, you need OPEN #3, SCR : MORE #3, TEMP : CLOSE #3.

You could supply the name in full, like N1_RAM2_THING. If the name is not recognised immediately MORE adds the DATA_USE default and tries again. You don't have to specify the full path in systems that use sub-directories, like the Thor and Miracle hard disks, Gold Card and the upgraded Atari ST Qdos Emula-

tor.

MORE implicitly uses channel #0 to display prompts and read control keys, so you cannot direct output to #0. The prompt window must allow input. It has an active cursor, so you can easily switch to and from MORE while multi-tasking other programs.

The prompt line starts with the current position in the file: a number between 0, the start, and the length of the file in bytes, which comes next. Both values match when the last page of the file is on display. The difference is the number of bytes 'below' the last time displayed.

It can be useful to spot task files when snooping around, so MORE shows the data space value stored in the file header if you use it to look into a task. The last part of the prompt line is the name, again taken from the file header:

853 of 65536 (8192) in DIY_Task

The example indicates that the 64K file DIY_Task uses 8K of data space, and 853 characters have been displayed. Once the first page has appeared you can move through the file with keyboard controls. When you have finished press ESC, or CTRL SPACE in task 0,0, to stop MORE and close the file.

PgDn or ALT down-arrow displays the next page. If you carry on at the end of the file MORE wraps back to the start. Down-arrow alone scrolls one more line onto the display. This is slower, but may be preferable if important text spans a page boundary.

You can move directly to any position in the file by pressing ENTER, typing the position, and ENTER again. The file will be displayed from the specified point; the first line will be incomplete unless the position chosen is at the start of the line.

If you type nonsense, or more than nine digits, the display is re-drawn at the current position. ENTER ENTER is a quick way to restore the display if it has been obscured by another task.

Backwards

ALT up-arrow or PgUp move backwards through the file. If you are at the start, it takes to you 512 bytes from the end of file, and displays from there. This quick route to the end of the file can be useful; for instance it reveals the names of extensions used by *Supercharge* and *Turbo* compiled tasks.

If you have just moved down a page, PgUp will move back to the previous page. If you want to go further ALT up-arrow and PgUp hop back in steps of 2048 bytes from the current position. The values 512 and 2048 are chosen by experiment; ideal settings are a matter of taste and window dimensions. They are stored as literal long words in the code file, and could easily be patched or re-assembled to different values.

The code is re-entrant, using no variables beyond the current task, so it can be used by several tasks at once. It dynamically adjusts to the space available in SuperBasic's buffer, reading large or small chunks from the file accordingly. Even if the buffer is small, the QL slave-blocks make access fast once the data has been found for the first time.

The main window display runs at the best speed possible. Several lines, or even the whole window, can be printed as one string, avoiding the overhead of Qdos calls for every line or character.

QL display output, measured in characters per second, is fastest if the main window used CSIZE 1,0 and the leftmost pixel x co-ordinate in the window is evenly divisible by 8. This is the default for SCR, CON and QL (F2) TV windows; (F1) monitor windows are faster without the ornamental borders of Sinclair and CST; remove these with BORDER 0; BORDER #0,0: BORDER #2,0.

MORE needs two windows; one must be CON-type, the other may be SCR or CON, or networked equivalents. Input can come from any file, or devices that support 'random access' system calls; this includes microdrives, ram disks, floppy and hard drives, as well as *DIY Toolkit's* MEM device.

If the MEM device-driver is loaded, MORE MEM displays the first page of system memory. You can view any address. Type ENTER 49152 ENTER to view the memory contents at the start of the Rom port. OPEN #3, MEM9_80p : PRINT #3, "Hello QL World" : CLOSE #3 creates a permanent buffer called MEM9, with a message inside; MORE MEM9 displays the contents.

At around 1K, MORE is a fat cat among DIY Toolkit routines, and calls Qdos literally dozens of times in each run. Unusually, it is more than just a building-block. You can customise MORE to suite your own system, using it alone or in conjunction with other commands.

The code could be optimised by register changes, branch and subroutine sequencing to save bytes, but it has been written to facilitate continuous development and customisation; I have not squashed the code to the minimum, because that would complicate further additions or changes.

Users can easily change the command name, messages, or step sizes, perhaps to overlap pages or suit favourite windows. Budding 68000 programmers might add code to indent long lines, alter the wrap-around, and clear windows before or after use. Many extra controls and options are possible: how about allowing negative entries to move any specified number of bytes backwards in a file?

The options are endless, with commented source available from DIY Toolkit. Please ask for Volume V for file Viewing (the name Volume M is already used for MultiBasic). Volume V includes the source and object code for MORE and *Quill* documentation. It

cost £7 on disk or your cartridge, from DIY Toolkit, Cwm Gwen Hall, Pencader, Dyfed, Cymru SA39 9HA.

The main assembler source listing has been divided into two sections. The first part appears this month, along with the hex loader for the entire code file. You can use the hex loader to create a working copy of the MORE command. In my next DIY Toolkit column I shall present the remainder of the assembler source code, and explain how it works.

Adding MORE

Listing Two is a simple Basic loader which reads the machine-code from DATA statements and stores it in a file. Lines 100 to 580 are the same for each DIY Toolkit project. Once you have created the code file, these commands will add MORE to your system:

```
X=RESPR (1024)
LBYTES "filename", X
CALL X
```

Listing one shows how MORE checks its parameters and generates the prompt message. This program is more complicated than most DIY Toolkit routines, and I have used some extra assembly-code features to keep the complexity under control. The code starts by declaring register and constant EQUates; these are symbolic names used to make the program easier to read, understand and modify.

The names BUF, OUTPUT_ID and PROMPT_ID correspond to 68008 register names. If your assembler does not allow EQU you need to replace the symbolic names with A4, D5 and D7 respectively.

The main block of equates defines offsets in the SuperBasic BUFFER area, used to store internal variables as the command is executed, and text en route from the file to the display.

All the variables are addressed relative to BUF, which is the offset of the buffer inside SuperBasic. This is initially 128 bytes long, and grows to fit long program lines. BUF comes immediately after Basic's system variables, so its start does not move relative to A6, the base address of Basic.

The 68008 instruction set makes it easy to read and store values at addresses indicated by two registers and a small offset. This addressing mode is ideal for access to tables inside Basic tasks — for instance, MOVE.W DEPTH (BUF,A6,L), D2 reads into D2 the word at offset DEPTH in BUF in SuperBasic.

Each offset in BUF is worked out from the one before, so you can infer the size of each area from the offset to the next one. Some areas are shared; for instance the space from FILENAME onwards is used to store the file name and the file header; later most of the file header space is used to store file text, at TEXTBUF. The calls to

DIY TOOLKIT

read WIDTH and DEPTH, or COLUMN and LINE, also store values in OLD_TOP, but these are not used and get overwritten later.

The routine labelled START is only used when you CALL the code file. It passes the name and address of MORE to SuperBasic. The value \$110\w indicates a word vector; other assemblers expect \$110 or \$110.w.

The next routine implements the MORE command. If there is more than one parameter it fetches an integer with CA.GTINTI otherwise it assumes #1. It picks up the ID of channel #0 as it reads the output window ID from the channel table.

The block labelled WINDOWS checks these IDs by turning on the cursor in channel #0 and reading the size of the output window with SD.CHENQ. These operations only work on display channels; other devices give a 'bad parameter; error.

From GET_NAME onwards the code assembles the file name. TRY_OPEN is a subroutine used to open the input file. MORE makes two attempts to open the file; first it passes the literal parameter value; if that fails it looks at offset 176 in the System variables for a pointer to the data default directory string.

This variable, SV.DATAD, is zero on standard QL systems. Toolkits and disk roms set it to the address of the default data directory string. It consists of up to 33

character, including an underscore at the end; it is set with DATA_USE and displayed with DLIST or PRINT DATAD\$.

If the parameter has no value MORE picks up the text from the Name List, like MultiBasic, in *QL World* March and July 1990. This means that you do not have to put the name in quotes unless it has a value in your program. CA.GTSTR is used to fetch the value of quoted strings, variables or expressions.

We arrive at OPENED if the file name is accepted. The next step is to print the prompt message in channel #0. The file size, task data space and name are all read with FS.HEADR. If the file header is not the expected size MORE assumes a MEM or other serial channel and skips the details, setting the notional 'length' to a few billion.

duce the need for device access, so MORE goes at healthy speed, even if the buffer is small or you run it from microdrives. If you want pages to appear all at once you can expand BUF by entering or loading along line of SuperBasic; the easy way is to PRINT "REM " & FILL\$ ("*",2000) to a file and MERGE it.

Source for the PRINT_ subroutines will appear in my next DIY Toolkit column, which will explain file input, keyboard control and display output. The next listing includes useful routines to read and write 32-bit long interger values, avoiding the slow and complicated floating point conversion that would be needed to use Qdos Rom Code. MORE can display file positions up to 4,294,967,295 - the limit of Qdos random access files.

Variables

At FIND_ROOM the code works out the size of the BUF area, allowing space for the variables at the start. The value in MAXTEXT is the number of bytes to be read from file each time. MORE is designed to work a screen at a time, so it reads no more than the number of characters that would fill the output window, calculated by multiplying DEPTH and WIDTH.

The QL 'slave block' buffers further re-



KEN BAKER SOFTWARE

32 Hunts Pond Road – Park Gate – Southampton – SO3 6QA – 0489 581056

All Programs for 768K Expanded QL with Trumpcard

SPEEDSORT

The ultimate sorter! Sorts arrays of any size by given element number. The speed is devastating – up to 80,000 sorts per second!

Machine Code £20.00

QLDL

QL Disk Library allows the user to catalogue up to 3000 program titles (around 75 disks at a time) in alphabetical order and cross-referenced by numbers given to the disks. No more searching for endless hours for that lost program.

Machine Code £15.00

SCREENMASTER

Screenmaster will record anything currently on screen in a 'squeezed' form, and reproduce the picture in a variety of cinematographic-style wipes and speeds.

Machine Code £20.00

PLANNER

One of the most useful programs you could wish for. It will provide a printout of dates, with day and week numbers, in a month-per-page format from a given starting date and number of forward days. It also includes routines for finding day and week numbers you can use in your own programs.

Machine Code £10.00

TRANQUIL

Tranquil converts Quill documents into BASIC PRINT statement form in order that they can be more easily manipulated for printing and adding printer instructions etc.

Machine Code £10.00

ALL PROGRAMS SUPPLIED ON TOP-QUALITY 3 1/2 INCH DISKS – POSTAGE & PACKING FREE

DBQL

In the third part, Tom Ashcroft looks at loops and selections.

So far we have looked at information retrieval mainly from the point of view of locating and displaying individual records in the database, but much of the value of computer databases lies in their ability to extract and collate information from many records. This power is most effectively realised by programming the database, and a later article in this series will consider programming in more detail. A fairly simple function, however, which most commercial packages have available from the keyboard, is the ability to make a selection from records in a database according to some specified feature or features. For example, a mail order firm might want to select all its customers living in Bristol to receive some promotional literature, and among those customers it might want to choose particularly, say, men aged less than 40.

Selection loop

To program this in DBQL is fairly simple and needs a loop to examine the appropriate field of each record in turn and select those which match with the given selection criterion. We have to decide whether to select-in those which match or select-out those which do not match, but if repeated selections are to be carried out with different criteria, it is better to flag the records that fail to match rather than those that do, so that selected records are left unflagged and available for later selection. The flag will be a letter S placed as the second character of field 0 of a non-matching record. (The first character place is reserved for the D deletion flag.) See listing twenty two.

The procedure first displays a list of field names and numbers and requests the number of the field to be matched using the new function, GETNUMBERS\$, detailed later, which ensures that only numeric characters are accepted.

Line 2610 requests the item for selection and the counter variable, selected, is set to 0. The loop in 2630 to 2690 examines the appropriate field of each record in the database and if the selection item is identified in the field the counter is incremented,

Listing twenty two

```

2540 REMark ***** SELECT
2550 DEFINE PROCEDURE se
2560 CLS:FOR j=1 TO fields:PRINT j,a$(0,j,4 TO
LEN(a$(0,j)))
2580 PRINT"Enter field number ";
2590 fldno$=getnumber$(1,fields)
2600 fldno =fldno$
2610 INPUT"Enter selection item ";selct$
2620 selected=0
2630 FOR j=1 TO nn
2640   IF selct$ INSTR a$(j,fldno) AND
a$(j,0,2)<>"S" THEN
2650     selected=selected+1
2660   ELSE
2670     a$(j,0)=a$(j,0,1)&"S"
2680   END IF
2690 END FOR j
2700 PRINT selected;" records selected"
2705 IF NOT selected THEN RETURN
2710 INPUT"Enter S to save selected records to
microdrive\" L to list them on screen\"Press
ENTER to return to menu ";q$
2720 IF q$="S" OR q$="s" THEN
2730   INPUT "Enter file name ";subfile$
2740   OPEN_NEW#6, "mdv1_"&subfile$
2750   PRINT#6,selected:PRINT#6, fields:PRINT#6,
chars:PRINT#6,0
2760   FOR j=0 TO nn
2770     IF a$(j,0,2)="S" THEN NEXT j
2780   FOR k=0 TO fields:PRINT#6,a$(j,k)
2790   END FOR j:CLOSE#6
2800 END IF
2810 IF q$="l" OR q$="L" THEN li
2820 END DEFINE se
2830 REMark ***** DESELECT
2840 DEFINE PROCEDURE des
2850 FOR j=1 TO nn:a$(j,0,2)=" "
2860 selected=0
2870 CLS:PRINT\ "Selection cancelled"
2880 END DEFINE des
2890 REMark ***** LIST
2900 DEFINE PROCEDURE li
2910 IF NOT selected THEN PRINT \ "NO SELECTION HAS
BEEN MADE":RETURN
2920 CLS:FOR k=1 TO nn:IF a$(k,0,2)<>"S" THEN
c=k:b_s_display
2930 END DEFINE li

```

provided the record is not already flagged, but the record itself is left unchanged. If a record does not match, the flag S is inserted in field 0.

After the loop the procedure prints the number of matching records, ie the value of selected. To make a further selection on a different field, the procedure is called again and this time counts only matching records not already carrying a flag, so that only records matching both criteria remain unflagged. A third selection would

leave only those satisfying all three criteria, and so on. At each stage the number of matching records is displayed.

Having made a selection in this way, what do we do with it? That will really depend on the needs of the database application but possibilities we could provide for are to display the selected records or to save them to microdrive as a separate subfile. Line 2710 offers these two options. Pressing ENTER alone at this point will terminate the procedure, while

entering S will cause the selected records to be written to a microdrive file. Option L calls a new procedure, Llist, which can also be called from the keyboard, and uses the B_S_DISPLAY procedure to print selected records to screen.

Some commercial databases have an arrangement whereby the database behaves as if it were composed only of the selected records and the remaining records no longer exist. This can be partially achieved by inserting the following line in the DISPLAY procedure

```
3015 IF a$(c,0,2)="S" THEN RETURN
```

so that only records without a selection flag will be displayed.

Another way would be to copy unflagged records to a separate array in memory and handle it as a separate file but this is not possible at present in DBQL, though it will become possible after the next article in the series when multiple file facilities are added.

Astute readers will have realised that last month's binary search provides a selection function of a sort, but it is applicable only to indexed fields and does not support multiple selections. Multiple selections would require reindexing of the selected records at each stage and this would probably cancel out the speed advantage of the binary search.

Having completed a selection procedure, the flags must be cancelled before an entirely new selection can be run and the procedure DESelect does this by setting the flag character to a blank in each record throughout the file. The flag, selected is set to 0. It is desirable to deselect the records before saving the file, otherwise the selection will still be in force when the file is reloaded.

Inputs in DBQL have so far been entirely open and no doubt users will have encountered error messages through accidentally entering inappropriate characters, for example entering a letter or punctuation mark when a number is called for. We shall now look at ways of error-proofing the program so that inappropriate entries are rejected.

There are three kinds of information we can store in a database viz characters (anything made up of letters or symbols), numbers and dates. As previously mentioned, all data are stored as strings and numeric characters are converted to real numbers only when required for calculation. Character entries have to be completely free to contain any printable character and little validation is possible, except to ensure that capital or small letters are used where this is essential. Four defined functions will now be described which check input of numbers and dates.

INTAKE is a defined function called when input of a number is required. It will accept numeric characters but rejects letters and most symbols by printing its own error message and waiting for the

user to try again. The only non-numeric characters allowed are a leading minus sign and a single decimal point. See **listing twenty-three**.

5100 and 5140 are simply to cancel the effect of the print 'follow-on' semicolons in 5090 and 5150 and avoid disruption of the screen display at the calling source.

Listing twenty-three

```
5050 REMark ***** INTAKE$
5060 DEFine FuNction intake$
5070 LOCAl j
5080 REPeat getstr
5090 wrong=0:decimal=0:INPUT q$;
5100 IF q$="" THEN PRINT:RETURN q$
5110 FOR j=1 TO LEN(q$)
5120 wrong=wrong+(q$(j)<"0")+(q$(j)>"9")-(q$(j)="."
AND decimal=0) -(q$(j)="-" AND
j=1):decimal=decimal+(q$(j)=".")
5130 END FOR j
5140 IF NOT wrong THEN PRINT:EXIT getstr
5150 PRINT\ "Enter number only: ";
5160 END REPeat getstr
5170 RETURN q$
5180 END DEFine intake
```

Listing twenty-four

```
5510 REMark ***** GETNUMBER$
5520 DEFine FuNction getnumber$(lo,hi)
5530 LOCAl q$,q
5540 REPeat getvalue
5550 q$=intake$:IF q$="" THEN q$="0"
5560 q=INT(q$):IF q>=lo AND q<=hi THEN EXIT getvalue
5570 PRINT "Invalid entry ";
5580 END REPeat getvalue
5590 q$=q:RETURN q$
5600 END DEFine getnumber$
```

The whole function is enclosed in a repeat loop, getstr. After the variable, wrong, has been set to 0 and the number has been entered as a string in 5090, the program checks for an empty string (caused by pressing ENTER alone) and if this is found, returns the empty string. Otherwise a FOR - ENDFOR loop is entered which checks each character in q\$ by means of logic statements which increment the value of wrong if the character is not a numeric character. (The QL is actually checking the Ascii code of the character rather than the value of the numeric character, see QL User Guide Concepts pp 5-9.) The presence of a minus sign therefore initially increments wrong but this is decremented again by a later statement if the minus sign is the first character in the string. A decimal point is dealt with in a similar way but also sets a flag decimal so that any subsequent decimal point coming later in the number will leave wrong incremented. If, at the end of the loop, wrong has not been incremented and still has a value of 0, then no unacceptable characters are present, 5140 exits the loop and 5170 returns q\$ to the calling source as a valid number. If wrong is not 0 then 5150 prints an error message 'Numbers only' and the loop returns to the input statement for another try. The PRINT statements in

INTAKE\$ alone is sufficient protection when entering numbers into a data field but in some other situations further local tailoring of the input is needed, for example, when choosing the field number at the beginning of the Select procedure above. Here only a positive integer between 1 and fields will do. A similar situation arises in other parts of the program and it is worth while writing a defined function to deal with it, such as GETNUMBERS\$ (**listing twenty-four**).

In this function a repeat loop, getfield, first calls INTAKE\$ to obtain an input. If an empty string is input a value of 0 is given to q\$. Then the integer of the input is taken, which eliminates any fractions, and finally a check is made that the input lies within the specified range, passed by the calling source as the variables lo and hi. If the conditions are met the loop is exited but otherwise a message is printed and the loop returns to the beginning, for a new input.

Dates are sometimes used in calculations and are better stored as numbers eg 25 12 90 rather than 25 Dec 90. In practice a six-digit numeric string will be used, in which the first pair of characters hold the day, the second pair the month and the third pair the year: DDMMYY. After the year 1999 the full year number will be needed, but that is some time away yet!

Listing twenty-five

```

5190 REMark ***** CHECKDATE
5200 DEFine FuNction checkdate$
5210 LOCAL j
5220 REPeat getdate
5230 wrong=0
5240 INPUT "(DDMMYY)";q$;
5250 IF q$="" THEN PRINT:RETURN "000000"
5260 wrong=wrong+(LEN(q$)<>6)
5270 FOR j=1 TO
LEN(q$):wrong=wrong+(q$(j)<"0")+(q$(j)>"9")
5280 IF wrong THEN PRINT "Invalid date ";:NEXT
getdate
5290 wrong=wrong+(q$(3 TO 4)<"01")+(q$(3 TO
4)>12)+(q$(1 TO 2)>31)+(q$(1 TO 2)<1)+(q$(1 TO
2)>30 AND (q$(3 TO 4)=9 OR q$(3 TO 4)=4 OR q$(3 TO
4)=6 OR q$(3 TO 4)=11))+(q$(1 TO 2)>28 AND q$(3 TO
4)=2 AND (q$( 5 TO 6))/4<>INT((q$(5 TO
6))/4)))+(q$(1 TO 2)>29 AND q$(3 TO 4)=2 AND (q$(5
TO 6))/4=INT((q$(5 TO 6))/4))
5300 IF NOT wrong THEN PRINT:EXIT getdate
5310 PRINT "Invalid date ";
5320 END REPeat getdate
5330 RETURN q$
5340 END DEFine checkdate$

```

Listing twenty-six

```

5350 REMark ***** UPPER$
5360 DEFine FuNction upper$(x$)
5370 LOCAL j
5380 FOR j=1 TO LEN(x$)
5390 IF CODE(x$(j))>96 AND CODE(x$(j))<123 THEN
x$(j)=CHR$(CODE(x$(j)) -32)
5400 END FOR j
5410 RETURN x$
5420 END DEFine upper$
5430 REMark ***** LOWER$
5440 DEFine FuNction lower$(x$)
5450 LOCAL j
5460 FOR j=1 TO LEN(x$)
5470 IF CODE(x$(j))>64 AND CODE(x$(j))<91 THEN
x$(j)= CHR$(CODE(x$(j)) +32)
5480 END FOR j
5490 RETURN x$
5500 END DEFine lower$

```

Listing twenty-seven

```

2330 CLS#4: PRINT#4,"Enter field names in order, up
to 8 characters in length."\"Enter Shift/ESC to
quit."
2335 PRINT "      Field name      Character, Number
or Date"
2340 REPeat fname
2350 INPUT \fields+1;" : "; field$;
2360 IF field$="@" THEN EXIT fname
2362 REPeat chektype
2364 INPUT TO 28;type$;:type$=upper$(type$(1))
2366 IF type$ INSTR "CND" THEN PRINT:EXIT chektype
2367 PRINT "Invalid type. Only C,N or D ";
2368 END REPeat chektype
2370 IF LEN(field$)>8 THEN field$=field$(1 TO 8)
2372 field$=type$&field$
2390 x$=x$&field$&"":fields=fields+1
2400 END REPeat fname
2410 PRINT\"How many characters in the longest
field? ":chars=intake$
2420 IF chars<11 THEN chars=11

```

There is no need to store separators, which are only required for clarity when printed. Dates are entered via the defined function CHECKDATE\$ (listing twenty-five).

CHECKDATE\$ operates on the same principle as INTAKE\$, using logic statements to increment the variable, wrong, when an erroneous condition is found. The repeat loop, getdate, first obtains an input and if an empty string is input this is returned as a default. Lines 5260 and 5270 check that the string consists of six numeric characters, and if wrong has been incremented at this stage 5280 prints an error message and goes back to the input.

With care

Line 5290 checks that the string is a valid date. The line is complicated and must be typed with great care since even a single mistaken character will cause malfunctions, which are sometimes quite bizarre. It contains a series of logic statements linked by "+" signs, each statement enclosed within brackets. In order as printed, the statements increment wrong for the following errors: month less than 1; month more than 12; day more than 31; day less than 1; day more than 30 in month 9,4,6 or 11; day more than 28 in month 2 when year is not exactly divisible by 4; day more than 29 in month 2 when year is divisible by 4.

If none of these conditions is true, wrong will remain at 0 and 5300 will exit the loop, otherwise an error message and a return to the input stage will follow.

Into caps

The defined function UPPER\$ will check any specified string passed by the calling source, and convert all small letters to capital letters. A FOR-ENDFOR loop checks the Ascii code of each character of the string in turn and if it is that of a small letter, subtracts 32 from the code to convert it into the corresponding capital letter, using the CHR\$ function. The function LOWER\$ converts upper case letters into lower case in the same way by adding 32 to the Ascii code when it encounters a capital letter in the string (listing twenty-six).

To put these functions into use the filename structure of the database has been modified and changes are needed in the existing procedures for entering and amending records. A\$ (0) has been changed to incorporate a flag in each filename which is used when entering data to indicate which of the defined functions is to be called. The flags are C (character), N (number) and D (date) and are inserted as the third character in each field of a\$(0).

Listing twenty-eight

```

1220 PRINT(a$(0,fld,4 TO LEN(a$(0,fld)))));TO
12; a$(0,fld,3);" : ";
1221 IF a$(0,fld,3)="N" THEN
1222 q$=intake$:IF q$="" THEN q$="0"
1223 ELSE
1224 IF a$(0,fld,3)="D" THEN
1225 q$=checkdate$: IF q$="" THEN q$="000000"
1226 ELSE
1227 INPUT q$
1228 END IF :END IF
1230 IF q$="@" THEN EXIT entryloop

```

Listing twenty-nine

```

1860 PRINT a$(0,fld,4 TO LEN(a$(0,fld)))));TO
11; a$(0,fld,3);" : "&a$(c,fld):PRINT TO 11
1861 IF a$(0,fld,3)="N" THEN
1862 q$=intake$
1863 ELSE
1864 IF a$(0,fld,3)="D" THEN
1865 q$=checkdate$
1866 ELSE
1867 INPUT q$
1868 END IF :END IF
1870 IF q$="" THEN :hold$(fld)=a$(c,fld):ELSE:
hold$(fld)=q$: reenter=reenter +(a$(0,fld,1)="I"):
END IF

```

(The first and second character places are reserved for the I flag and the value of active respectively.) The fieldname proper now begins with the fourth character in a\$(0) and, allowing 8 characters for the name, requires a minimum value of 11 for chars. CReate has been altered as follows to enable the flag to be input after each fieldname (listing twenty seven).

Several new lines have been added to procedures EN and AM, calling the appropriate function according to the flag for each field in turn.

ENter is modified as in listing twenty eight. If the ENTER key is pressed as an entry, a default value of 0 is entered for a number field or 000000 for a date field.

AMend is changed similarly (listing twenty nine) but default values are not given as ENTER alone here means 'leave unchanged'.

Next month

It is also necessary to change each reference to 'a\$(0,3 to..)' to 'a\$(0,4 to..'. These occur in lines 2076, 2077, 2083, 2086, 2266, 4070, 4150, 4380 and 4980.

Next month we shall convert DBQL from a flat file to a fully relational database by adding facilities to have more than one file open at the same time.

TF Services

Qualsoft Terminal

Viewdata/VIS2/V100 QL TERMINAL EMULATOR

Multitasking for electronic mail, PRESTEL etc, CET download, Phone directories for ALL (yes ALL) modems for the QL, autodial (where poss) and logon, two-way FILE TRANSFER to ATARI ST, IBM PC, Psion Organiser (via comms link), XMODEM, real time clock/timer, buffered logs to file/printer, transmit files, editable command line, Swedish/Norwegian/German options, translates, editor etc.

SOLVES ALL PACKAGED MODEM SOFTWARE PROBLEMS

Unbuffered modems usable with Miracle Modemport

Software (3.5" or ndv) & manual £30

COMPUTER CLEANER

HELP STOP POWER SUPPLY INDUCED LOOKUPS AND PROTECT COMPUTERS

Much improved mains filters, 40-80 db cut. Contain inductance, spike filters (three in 3 & 4 way) and certified 240vac capacitors.

2-way (5a adaptor)..... £14

3-way (5a adaptor)..... £18

4-way (13a trailing socket)..... £24

MINERVA

The ULTIMATE operating system upgrade

MKI1 MINERVA now with battery for 256 bytes ram, DRASHPROOF clock & interface (12C) for robots etc. Can autoboot from battery ram.

UPGRADE MKI1 now to latest rom. 1.8n FREE.

Others £5 (send chip & utilities disk)

Original MKI..... £40 MKI1..... £65

MIRACLE SYSTEMS GOLD CARD COMPATIBLE

All prices include VAT and postage in the UK

QL SPARES

Faulty QL boards (excl plug in ICs)..... £9
Keyboard membrane... £8 Circuit diags... £2
68000 cpu..... £8 1377 PAL..... £3
J11 rom set..... £10 Power supply... £12
8382 ULA..... £10 8381 ULA..... £10
8849..... £8 MDV ULA..... £12
Other components (sockets etc).... pls PHONE

BULLETIN BOARD

QBBS - UK's first QL scrolling board.
QL Echomail around Europe, and FIDO netmail
Around 10mb of files including the WHOLE
King James' bible!

071-706 2379 V21/23/22/22bis 24hrs

QL REPAIRS

QLs tested with Thorn EMI test rig and ROM software

(MDV hardware extra if required)

£27 - 6 month guarantee

FILE TRANSFER

Programs to transfer text AND programs error free QMODEM between computers

Available for IBM PC and compatibles, ATARI ST and Sinclair QL. Connects to Psion organiser via Psion comms link

Qualsoft program (per m/c)..... £7.50

Serial lead (name 2 computers).... £10

Complete package for 2 computers. £25

MAIL ORDER ONLY

12 Bouverie Place, London W2 1RB (tel: 071-724 9853)

SCROLLING BBS FOR ORDERING/QL HELP - 071-706 2379 (all speeds to V22bis)

Fax: 071-706 2379 Prestel: 017249853

★ sinclair QL ★

EEC LTD MAIN SUPPLIER OF
Sinclair QL COMPUTERS & PRINTERS
QLs FROM £125 PRINTERS FROM £130 inc VAT

The expandable system for small businesses, beginners, and experts.

QLs COMPLETE JM ROM FULLY TESTED AND WITH 6 MONTHS WARRANTY

Package includes PSU TV lead, QL Software 2.35, Quill - Word Processor, Abacus - Spread

Sheet, Archive - Records, Easel - Business Graphics £125

Customers buying one of the QL Computers are given one year's free membership to QUANTA

(Help, Newsletters, & 400+ Library of Programs - mostly free).

QUANTA MEMBERS can obtain a £5 discount

As above but fitted with JS ROM

£150

★ £50 PART EXCHANGE ★

OFFER ON QL

Back up QL with JM rom £75. JS rom £99 consisting of keyboard unit, original Psion software wallet, leads and (see IN EXCHANGE FOR YOUR OLD QL (if you want to keep it add £10 to the above prices)

Spectrum Plus or 128K (working or not working as long as repairable), power supply for QL, if required £12, instruction book £5.

PC/QL KEYBOARD AND INTERFACES

PC AT 102 KEYBOARD AND CONNECTOR £25

QL TO PC INTERFACE FOR ABOVE £75

CASE AND LEAD FOR EXTERNAL MOUNTING-

TO ENABLE BOTH KEYBOARDS TO BE USED £17

OUR PRICES NOW INCLUDE 17.5% VAT

AND HAVE NOT BEEN INCREASED

QL Psion Software 2.35	pack £20	Quill, Abacus, Archive, Easel each	£12
QL Power Supply Unit	£12	Joysticks with QL lead	£12
Centronics Printer Interface	£27	Keyboard Membrane	£9.50
Complete User Guide	£8	QL ICs and Spares	ASK
Cartridges, Wallets of 4, new	£12	4 unused prog cartridges	£9

20 new cartridges in plastic storage box £55

SOFTWARE QL Decision Maker £18. Home Finance £18. QL Games £9. Q-Liberator

Super Basic Compiler (disc) £60. Q-Liberator Budget Version for (MDV) £30. Q-Load

Superbasic Quickloader with Q-ref Programming utility (Mdv) £15. File Transfer Programs.

QL to other systems including PC: Text Tidy 315. Discover £20 Multi-Discover £30

BUSINESS SOFTWARE (Suitable for standard QLs) Small Traders Pack £24.95. Invoice

£19.95. General Ledger £19.95. Set of three £50. Mail Merger £14.95. Filter Pack £14.95. Also

Stock Accounting £39.95. (For expanded QLs). Send SAE for software leaflets, or phone



All prices include VAT. Terms CWO. Access or Visa Minimum order £10.
Carriage £8.00 for printers, monitors & QL (overseas £20.00) other items £3.00 (overseas £6.00)

EEC LTD

18-21 MISBOURNE HOUSE, CHILTERN HILL, CHALFONT ST PETER, BUCKS. SL9 9UE

Tel: 0753 888866 Fax: 0753 887149

SOFTWARE FILE

TRANSFER UTILITY

INFORMATION

Program: *Transfer Utility*
Price: £9.95 (plus 5/10% for overseas buyers).

Supplier:
 Digital Precision Ltd.,
 222 The Avenue,
 Chingford, London E4 9SE.
 Tel. (081) - 527 - 5493.

Brian Davies tries on a no-frills conversion file for disk and microdrive.

programs have been sold in the past, but they may not be easily obtainable now.

Facilities

Transfer Utility is a fairly new program, supplied by the largest QL software house, and priced very reasonably. It is not a simple mdv-to-flp converter, and provides extra facilities which will make it of general interest. In fact, as the name suggests, the basic purpose of the program is to transfer files from one medium to another, but Digital Precision clearly feel that the program will appeal particularly to microdrive users who are converting to disk as the program is supplied only on disk.

By Digital Precision standards, the instruction manual is minuscule - three sides of text. It is supplied in both paper and disk file form, and is very easy to read and understand. The program itself is straightforward to use. It is only when one gets ambitious, and makes use of the extra facilities, that serious thinking is desirable. The two obvious features are the facility to perform search-and-replace operations on *any* strings, and the ability to copy *all* files, not just those which existing commands such as WCOPY will recognise.

The last point needs clarifying. Some users will believe that WCOPY can copy any file from one medium to another,

but this is not true. This command is 'blind' to certain file names and will ignore files which have them. The same is true of copying functions with some other programs, such as *Files II* (part of *Taskmaster*).

WCOPY

The point should not be over-emphasised, since many users would not need to make copies of files with unusual names, but it is worth giving examples of what *Transfer Utility* can copy that WCOPY and the *Files II Backup* command can't handle. Three copies of a test disk apparently produced 11, nine and three files respectively, using the three programs/commands mentioned. The test disk appeared to have 10 files on it when checked with the DIR command from *SuperBasic*; the space taken on the test disk was 474 sectors. The copy produced by *Transfer Utility* appeared to match the original, in all respects. The disk produced by WCOPY apparently held only two files when checked from *SuperBasic*, taking six sectors. In addition, one of the files initially copied was overwritten later, and it was necessary to repeatedly press the A key to persuade the command to continue with (unsuccessful) attempts to copy several more files. In short, WCOPY seemed baffled by the test disk.

Files II displayed only three files during copying, and three files during copying, and three were shown when the *SuperBasic* DIR command was used, the space taken being nine sectors. Of the files on the test disk, wight had names

The microdrive cartridge is still far from dead, but many users turn to disk drives at some stage during their life with the QL. If you have a program such as *The Editor*, it is not too difficult to alter existing microdrive program files to make the programs run from floppy disk, by using the search-and-replace function to replace all instances of 'mdv' by 'flp'. For users who do not have a suitable editor, or who do not feel competent to perform such operations, various conversion

```
SPECIFY DEVICE TO COPY FROM (default mdv1_):          mdv1_
SPECIFY DEVICE TO COPY TO (default flp1_):             flp1_
SPECIFY DEVICE TO USE AS A TEMPORARY STORE (default flp1_): flp1_

Ensure that the source medium is in mdv1_.
Ensure that the destination medium is in flp1_.
No medium should be write-protected.
SHOULD flp1_ BE FORMATTED? (Y/N)N

SPECIFY TRANSLATE FROM : mdv
SPECIFY TRANSLATE TO   : flp

SPECIFY TRANSLATE FROM : flp2
SPECIFY TRANSLATE TO   : flp1

SPECIFY TRANSLATE FROM :
SHOULD THE MATCHES BE EXACT (aBc=aBc) / SEMI-EXACT (aBc=ABc) / INEXACT (aBc=Abc)?
(E/S/I)S

Transfer boot? (Y/N/A/Q)
```

The screen: waiting selection for transfer.

which contain a 'splodge' character, and one had no *displayed* name. As noted above, the blind spots of other copy routines may not be of consequence to many users, but there are quite a few commercial program disks which contain files with unusual names (often as part of a copy-protection effort) which would not be copiable in the usual fashion (and the programs might, therefore, not run from the copies). The program on the test disk did not run when copied by WCOPY and Files II.

Protected

It has to be made clear that Transfer Utility is not intended to be a means of copying copy-protected programs; for instance, identical copies will not be made of programs which are protected by the requirement for a master disk to be in a particular drive, nor of programs which have the device names identified by non-consecutive characters within program files. An attempt to copy the cartridges from the old Quest program *Cash Trader* was unsuccessful, for example, as the program would not run. Specific examples of files which can be handled are those of zero length, or with zero-length file names, and those with file names containing null characters.

With a small number of files, copying speed may be of little consequence, but it is worth recording that Transfer Utility was considerably faster in the test described than WCOPY, and this might be a useful factor when copying large numbers of files and media.

Strings

The string-conversion function permits up to 32 substitutions to be made during the transfer process. Each substitution can involve up to 64 characters in the strings concerned, but the replacement must be of the same length as the original strings. The substitutions are done in succession; that is, after a file has been processed for one, it is then processed again from the start if a second (or subsequent) substitution has been specified. The specified substitutions and

```
SPECIFY TRANSLATE FROM :
SHOULD THE MATCHES BE EXACT (aBc=aBc) /SEMI-EXACT (aBc=ABC) /INEXACT (aBc=AbC)?
(E/S/I)S

Transfer boot? (Y/N/A/Q)A
Copying boot of length 62 with 2 (=2+0) translates
Copying _ of length 3584 with 6 (=5+1) translates
Copying ident of length 5 with 0 translates
Copying menu of length 14744 with 9 (=7+2) translates
Copying security of length 40344 with 21 (=13+8) translates
Copying create of length 19352 with 21 (=13+8) translates
Copying group_exm of length 285 with 0 translates
Copying analysis_exm of length 3807 with 0 translates
Copying vatfile_exm of length 363 with 0 translates
Copying clone of length 1961 with 5 (=3+2) translates
Copying of length 12184 with 8 (=6+2) translates

11 files totalling 96691 bytes transferred in 78 seconds
with 72 (=49+23) translates.
FINISHED... WANT ANOTHER TRANSFER? (Y/N)
```

The screen: transfer process now completed.

strings are applied to all of the files on the disk. Talking in terms of device names may obscure the fact that it is possible to make replacement of text strings which are nothing to do with such matters. For instance, you might have a long text file in which the word 'close' needed to be replaced by 'exact', and this can be done during the copy process.

Reset

The supplied disk contains three files of extension, loaded through the RESPR command, making it desirable to boot the program after a reset. Replacing the RESPR by the ALCHP command (from a Toolkit) in the boot should make it possible to run the program after other programs. For those users who feel they can't do quite what they want with the program as it is supplied, the SuperBasic source code is provided on the disk, and can be modified as desired (but not sold). A Turbo-complied version of the program is run from the boot file. Not surprisingly to use the program itself to make a backup copy of it.

The screen presentation during use is simple, and easily understood. The source and

target devices have to be specified (default are mdv1_ and flp_ respectively), as does the device to be used for temporary storage. If a ramdisk is used, you are asked if the disk interface in the system is of the MCS type, as the program needs to know this information because the MCS interface has a non-standard way of dealing with ramdisks. Changes to the specified devices are retained for when the program is next run. The option to format the target medium is given.

The 'from' and 'to' strings for each conversion are then specified the ENTER key alone being pressed if no conversion are required. To avoid awkwardness where both upper- and lower-case characters are involved on conversion strings, there is a choice of three methods of dealing with case - 'exact' being of the form aBc=aBc, 'semi-exact' being aBc=ABC, and 'inexact' being aBc=AbC. Once conversion commences, the file names, sizes and number of conversion made in each file are shown on the screen. The usual Y/N/A/Q choice is offered before copying takes place.

Figure two shows the screen at the point where the program is awaiting the user's selection

for transfer of the (first) file 'boot'. The total space taken by the copied files is shown upon completion, along with the time taken and the total number of string conversions. The option is then offered of going ahead with another transfer operation. Figure one shows the screen when the transfer process has been completed; the program is offering the option to do another transfer operation.

Unseen

This is a no-frills program, which does the job quickly, without placing heavy demands upon the user's computer knowledge. It is not the kind of thing one would need every day, but it is cheap enough to buy for occasional use. For users converting several programs from cartridge to disk, the price may be low enough to be justified for only the one session. The ability to 'see' files which are unseen by other copy commands could be of considerable use to the more technical user. It is good to see Digital Precision introducing more low-cost utilities, as there has been a dearth of such things since PDQL disappeared from view.

SOFTWARE FILE

INFORMATION

Program: *Discopy* 2.06
Price: £7.50
Supplier: Qfile
Apartado 2110
P-1103 Lisboa
Codex
Portugal

DISCOPY

From the makers of MS-Qlink, this new disk copying utility works well and is more versatile than simple COPY or WCOPY, as Bryan Davies finds out.

Programs don't come much cheaper than this one. It is, perhaps, in the mould of those PDQL used to sell, in that it is a single-function program which could be useful to users who do not have (maybe can't afford) any of the file utilities which incorporate several functions. As its name implies, *Discopy* is a program for copying disks. If it made only straight copies of files on Qdos-format disks, there would seem to be little point in having it, but there is more to it than that. The supplier of *Discopy* - Qfile - also sells *MS-QLink*, which is a similar program, but has more functions.

Fixed

That program was reviewed last year, and the conclusion was that it had good features but suffered from some problems, these subsequently being fixed to give an unusual, and useful, utility. A common feature of the two programs is the ability to handle disks of other formats besides Qdos. Specifically, *Discopy* handles 'any QL, MS-DOS and Atari TOS disk (720K, 360K, single- or double-sided). The majority of QL users presumably have just the one computer, but a sizeable minority are in 'multi-computer families', and the Atari and (IBM-type) PC seem to be the usual other machines.

The program will run in the background, and uses ram to reduce the amount of disk-

swapping required. It can be used with either single or dual disk drives. A memory expansion is not necessary, but there

will be more swapping of disks if memory is limited. Disk interfaces which do not permit direct disk sector addressing

are not suitable for use with *Discopy* (or with various other programs); the user is advised to purchase updated eprom

```
#1 May 88 16:59:02                                caps off / memory = 24678

An extra definition of DIR zapped.
An extra definition of EXEC zapped.
An extra definition of EXEC II zapped.
An e F4 Discopy 2.06 ESC
An e
An e Copy Report
An e Origin Aplt Target file2
An e Trks + 4 = 80 Size
An e
An e
An e
An extra definition of NEW zapped.
An extra definition of CLEAR zapped.
An extra definition of OPEN_IN zapped.
An extra definition of OPEN_NEW zapped.
An extra definition of CALL zapped.
An extra definition of CONTINUE zapped.
An extra definition of RETRY zapped.
An extra definition of PROG_USE zapped.

We have finished.
```


chips for MicroPeripherals or Medic interfaces. The program was written in SuperBasic and compiled with Turbo.

Many users will be familiar with (and annoyed by) the collection of additional commands, which accumulate in the QL as additional application programs are loaded. When these extension commands are listed, some of them may appear several times. This can cause problems; at the least, space is wasted. It would be nice to have some sort of filtering process, which would get rid of duplicate commands. Discopy has such a function (as does MS-QLink), and its action can be seen in the illustration, partially obscured by the main menu.

Two parts

The program boot is in two parts, the second one being called only if the first decides that the particular QL being used has the AH or JM Qdos version. When the second boot runs, it calls a task that looks through the commands and 'zaps' duplicates. As can be seen on the illustration, this can get rid of quite a lot of unwanted items (apparently, nearly 10KB of them from my system). The screen dump shows only the one 'page' and there were a similar number of commands on the previous page, making of the order of forty that were removed.

Discopy itself uses a run-time version of Digital Precision's Xtras file, which is similar to the Turbo_TK_Code file that is automatically loaded by the boot routine on my system. The Discopy instructions point out that the supplied boot can be modified to remove the line which loads the extensions, if the user already has such a line in an existing system boot; starting via the second boot file, instead of the first, by-passes the loading of extensions. Discopy itself can be started with EXEC or EXEC W and will run alongside other common programs without conflict. About 20 KB of memory is used by the program when loaded, and a further small amount is taken during copy operations.

Instructions are provided in a small, well-printed booklet.

Adequate information is given to meet the needs of most users. The single menu is fairly self-explanatory. Source and target drives can be toggled in the range flp1-4, the number of disk sides can be toggled between 1 and 2, and the number of tracks can be set to 40 or 80, or anywhere in the range 1-85 if the + key is used. F4 redraws the menu display, ESC will take you out of the program, or back to the menu after (some) messages are displayed.

The Format option is initially not active; that is, it is in green on the menu, as opposed to the black of the other options. To have the target disk formatted before a copy is made, you press F, which turns the colour of Format on the menu to black. This is out-of-line with the manner of selecting the other options, as you would expect to have to press F for the format to be done, rather than simply to make it possible to do a format after a further keypress. In practice, selecting C to make a copy then causes the target disk to be formatted, but no copy is made; you have to select C again to get a copy.

The behaviour of the program after Format was selected was rather baffling, in fact. Having obtained a format by selecting Format and then Copy, a check was made of what was on the target disk, by stepping-out to SuperBasic and doing a DIR. On using F4 to refresh the menu after going back into Discopy, the Copy process started, without the display being refreshed. It appeared that any other key (including ESC) would start the Copy process, also. The menu routines need correcting in this area, but the standard format and copy operations were performed successfully.

Format

The 'Trks' options relates only to copying; you cannot change the format mode, which is always the standard 80-track one. You can, however, format disks 1- or 2-sided. This operation wasn't very successful on my system; although a message indicated a 720-sector format was completed, after '1-sided' had been selected, and the Copy process went ahead, the result on several occasions was

a target disk that caused the system to lock up when a DIR was done. In case particular features of my loaded software were causing problems for Discopy, the latter was booted from a 'clean' machine, but the message 'disk. . . in use' appeared each time a Copy was requested, and it wasn't possible to proceed with any operation.

There may be some conflict here relating to the JM rom, although the program appears to detect that, since it goes ahead with the task of deleting surplus copies of extension commands. Typing individual boot lines in separately enabled the program to start, and work normally.

Report

The Report option may be a bit too technical for some users to understand. It is normally switched-on, by default, and the progress of a Copy operation is shown on the menu by Track counters for source and target disks. The 'Tracks' figure for the source disk will be larger than that for the target (until a copy is completed), and depends upon the amount of ram available to the program. From 20 to 80 tracks were read before writing started in the tests. While the Report function is on, the program does not attempt to re-read or re-write from/to disks when there is an error. That is, the copy operations stops if an error occurs.

If Report is switched off, as is suggested for when a copy is done as a background job while the user gets on with something else, there is no report of the progress of the copy, and the program makes up to four attempts to read/write information when an error occurs. You are also advised to switch Report off when using QJump's Ptr_Gen file, as Discopy will otherwise lock up when run as a background job with the pointer interface.

There is a configuration routine which can be used to alter the default settings. The empty area at the bottom of the menu box is used to display program messages, such as requesting confirmation that a format should be done. On completion of one Copy, you can immediately do another with the same

source disk. If the source disk has a volume label, the target disk is given the same label, which could lead to confusion later on. As an aid to 'normal users', the numbering of tracks and sides is altered by the program. That is, instead of being presented with numbers that don't seem too sensible to the average person, you get 1 to 80 for tracks and 1 to 2 for sides (the usual is 0-79 and 0-1). As sector numbering is 'normal', it is possible to test the program's ability to copy them, but it worked satisfactorily on several Qdos disks, and with three MS-DOS disks. There was some problem with two other MS-DOS disks used for test, but, after they had been subjected to a 'revive' operation on the PC, they were copied alright by Discopy, suggesting that there had been (previously-undetected) errors on these disks. It was quite interesting to have errors on 'foreign' disks pointed out in this fashion. The TOS format is similar to the MS-DOS, and there is no reason to think the program cannot handle it as well.

Quirks

Apart from the quirks mentioned, the program worked properly. The current version is 2.07, but the only reported change in that from the review version was made to correct a problem occurring only with the Atari ST QL emulator. It might not be obvious to some people that what the program is doing is not simply copying files from one disk to another - which can be done with the COPY or WCOPY commands - but producing a 'replica' disk, with the data located in the same place as on the original. For example, copying the working MS-DOS program disk supplied with the *Conqueror PC* emulator produces a disk that can be used for booting MS-DOS; if the DOS Copy command were used instead, the resulting disk would contain all the files but they would not be correctly located on the disk to allow it to be used for booting. This function can be handy, and is normally available only as just one part of utility programs which cost more than Discopy.

SOFTWARE FILE

INFORMATION

Program: *QL Bargain Software Pack: Inkwell Deluxe, Inktyper, Cuewell 2 and 20 extra fonts.*

Supplier: *Rob-Roy Software, 94 Teignmouth Road, Clevedon, Bristol, BS21 6DR.*

Price: £10.00 for 3 1/2 or 5 1/4 in disk. Send two mds with order for micro-cassette copy.

Now here is a bundle of software well worth every penny of the ten pounds asked.

I am sure that most readers in the past will have spent far more on the purchase of programs than this, only to find themselves bitterly disappointed when they start to use them. With this software, I assure you that you won't.

This package contains *Inkwell Deluxe* and *Inktyper*, a combined program which gives you access to literally hundreds of print styles for instant use on your printer. In addition there are *Cuewell 1 and 2*, two front end programs which act as disk and mdv management systems.

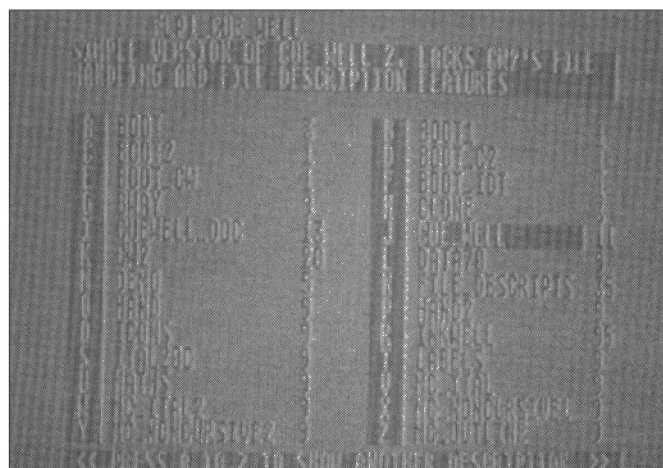
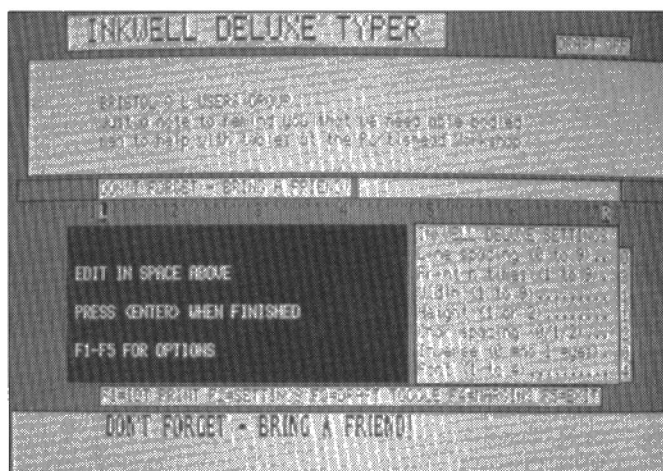
Inkwell Deluxe is a combined font editor and print utility. Thirty complete fonts are provided, each of which can be manipulated to individual requirements. They can be shadowed, rotated, flipped, outlined, italicised, emboldened and shrunk, in normal or italic print.

You can print documents using any of the fonts, with control over line spacing, print emphasis, character spacing, character width/height and print density.

It works with all Epson compatible dot matrix printers (and to a limited degree with the Serial 8056). It will work with

QL BARGAIN SOFTWARE PACK

John Shaw tries 'the bargain of the year so far' for the unexpanded QL.



any standard text editor which can produce Ascii files, including of course *Quill*.

Let us say, for example, that you have a *Quill.doc* which you would like to have printed

in different fonts with varying sizes. All you do is to make the first line of the *.doc* as follows:

##

Then enter control codes for

the different fonts on the *.doc*:

{W3H2U2L0C2}

Translated, this means: Width 3, height 2, using-font 2, line spacing 0, character-spacing 2.

Note that curly brackets are used { } to enter the codes.

Then you change the fonts, etc. as you progress through the *.doc* and you finally finish with:

{##}

When you have saved it, you then run it through the *Inkwell* program and it gets printed as you have planned it. There is a previewing facility to make sure you like the layout.

The *Inktyper* part of the program is a natural development of *Inkwell Deluxe*.

This now allows your QL to be used as a very versatile electric typewriter. In a flash you can be printing direct to your printer from the QL keyboard using any chosen font.

The display in front of you emulates a normal typewriter including such things as margin settings. As you type you still keep a view of the previous six lines of print. In addition, by pressing F1 you can see a representation of the style of print and actual line length before it goes on paper.

There is draft mode for a quick run through to check all is well. Then there is the nlq mode whereby all the print styles and other features you have put in are activated.

A comprehensive manual is supplied to help with any

SCREEN ECONOMISER

No burn-outs with a user-friendly screen timeout program.

INFORMATION

Program: Screen Economiser.

Supplier: Dilwyn Jones Computing, 41 Bro Emrys, Tal-y-Bont, Bangor, Gwynedd.

Price: £10.00. Available on Mdv, 3 1/2 or 5 1/4 in disk for the unexpanded QL.

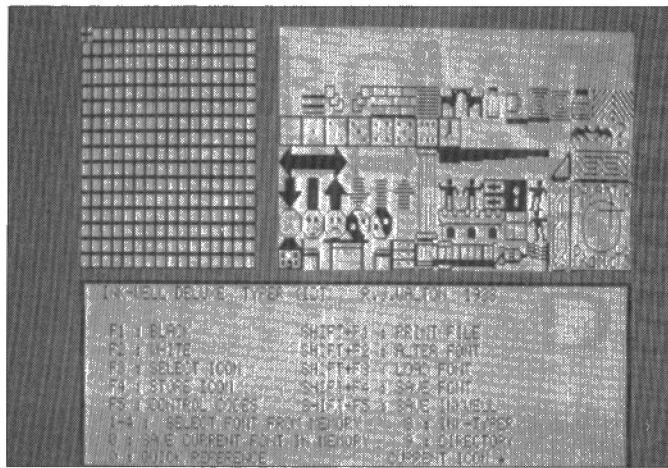
Estournet has produced a handy little routine which you load as a BOOT file for your program, and if the screen doesn't change its display for a preset period of minutes, it shuts down the screen display. It doesn't switch off the QL.

To recover your screen you just touch the <space> bar and as if by magic it reappears again.

You decide how long a delay you would like to have (the default is 10 minutes) and set it in the BOOT by inserting the word SECOVAL followed by a number between 1 and 255, as in SECOVAL 15. The number inserted represents minutes.

This is a simple program, easily set into motion and valuable to those people who are inclined to fall asleep in the middle of a late night session!

Did you know that if you left the display fixed on your screen then after a while it will burn the phosphor coating? I didn't. I think of the number of times I have left a Quill_doc on my screen for hours having answered the door to an unexpected visitor. Now help is at hand. Gerard



unusual printer codes which may be needed.

Cuewell and Cuewell 2 are two friendly front-end systems so designed as to make all your flp and mdv housekeeping operations as simple and easy as possible using a minimum of keystrokes. Cuewell is the simpler version which has been updated by Cuewell 2. The full blown Cuewell 2 shows an alphabetically sorted directory of up to 156 files, with file sizes.

You can Copy, Load, Format, Rename and Delete files with a single key press. In addition, Cuewell 2 allows you to store

information about each file and add notes, which will save you loading the .doc or any other files, to see what they contain.

Possibly the greatest benefit is that it sits in high memory, so that even if you type NEW it still remains usable.

The whole directory of a disk or mdv can be printed with one key press, so you can print out details of your whole disk library in minutes, what a facility!

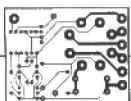
Any of these programs is worth the £10.00 asking price. For four, it has to be the software bargain of the year so far.

Qtris *Lear* DATA SYSTEMS

A SUPERB RENDITION OF THE ALL TIME FAVOURITE GAME NOW ON THE QL

- One or two player option (simultaneous play).
- Smooth, fast and colourful graphics.
- 100% hand crafted machine code.
- All the features of the original and MORE.

Only
£11.95 INC.



PCB CAD

THE STANDARD PCB DESIGN SYSTEM FOR THE QL

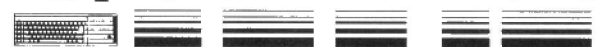
- A 32 by 32 inch drawing area.
- Resolution down to .001 of an inch.
- Up to 16 separate layers with variable colour assignments.
- 16 predefined Pad shapes and sizes.
- 64 Track widths from .005 to .32 of an inch.
- The number of components on a layout is limited by memory only.
- Auto repeat on all elements - ideal for memory planes.
- Can be used for surface mounted devices.
- Auto via system for multilayer boards.
- Over 32,000 user definable library components.
- Can output plot files to any valid QL/THOR device.
- Produces Dot matrix, HP-GL or Postscript laser files.
- Supports Trump card and ABC Elektronik keyboard interface.

Now Only
£69.95 INC.

Minimum system requirements: 512K expansion and 1 disc drive.

Please make cheques payable to Lear Data Systems and send to:
Lear DATA SYSTEMS
6 SOUTH VIEW GREEN, BENTLEY,
IPSWICH, SUFFOLK IP9 2DR.
TELEPHONE: (0473) 310804

QUANTA



Independent QL/Thor Users Group

Worldwide Membership is by subscription only, and offers the following benefits:

Monthly Newsletter - up to 40 pages

Massive Software Library - mostly FREE!

Free Helpline and Workshops

Regional Sub-Groups. One near you?

Advice on Software & Hardware problems

Discounts from most major suppliers

Subscription just £14 for UK members

Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

* Now in our EIGHTH successful year *

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex. SS7 4JD
Tel (0268) 754 407



SUBSCRIBE TO *Sinclair QL World* AND GET THIS GREAT BINDER...



...ABSOLUTELY FREE!!

Fill in the coupon below and send it with your remittance to: **MSM**
Subscriptions Department, Lazahold Ltd, P.O. Box 10, Roper Street, Pallion
Industrial Estate, Sunderland SR4 6SN. Telephone 091 510 2290.

(The first issue of a new subscription to be delivered will be one or two issues after the one you placed your order in.)

Please start/renew my 12/24 month subscription to SINCLAIR QL WORLD and send my free binder.
I enclose my cheque/money order for £ _____ made payable to MCPC Ltd or debit my Access/Visa card.
No _____ Expiry date _____
Name _____ Address _____

(Please enter postcode to ensure prompt delivery)

Signed _____ Date _____ SQL 0891

RATES	UK £23.40 12 mth	Europe £32.90 12 mth	Rest of £40.90 12 mth	OVERSEAS RATES INCLUDE AIRMAIL SERVICE
	£46.80 24 mth	£65.80 24 mth	World £81.80 24 mth	